EXACT ALGORITHMS FOR ROUTING PROBLEMS

A Dissertation
Presented to
The Academic Faculty

By

Felipe Lagos

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Industrial and System Engineering

Georgia Institute of Technology

December 2019

EXACT ALGORITHMS FOR ROUTING PROBLEMS

Approved by:

Dr. Natashia Boland, Advisor
School of Industrial and Systems Engineering
Georgia Institute of Technology

Dr. Martin Savelsbergh, Advisor
School of Industrial and Systems Engineering
Georgia Institute of Technology

Dr. Alejandro Toriello
School of Industrial and Systems Engineering
Georgia Institute of Technology

Dr. Alan Erera
School of Industrial and Systems Engineering
Georgia Institute of Technology

Dr. Michael Hewitt
Quinlan School of Business
Loyola University Chicago

Date Approved: August 14, 2019

To María Francisca

ACKNOWLEDGEMENTS

I would first like to express my gratitude to my advisors, Martin Savelsbergh, Natashia Boland and Alejandro Toriello. Working with them has been a privilege. I am thankful for their guidance, support during my studies, dedication and patience. Their continuous encouragement have been fundamental for my development as an independent researcher. Thank you all for spending countless hours challenging me to become more rigorous and clear in my research.

I would also like to thank the members of my thesis committee for their service: Alan Erera and Michael Hewitt. I am truly lucky to have had such experts in transportation and logistics exposed to my research.

Thanks to all my friends made in Atlanta, in particular to ISyE PhD students for the time spent together doing research, studying or just chatting over a cup of coffee. A special thanks to Mathias Klapp with whom I spent several hours working on the first paper and who was a significant support during my first year of the program.

The biggest acknowledgment goes to my dear wife María Francisca, who four years ago joined me in this adventure. These have been unforgettable years full of experiences I will never forget. I am grateful for the unconditional support, during good and bad times, always there. This thesis would not have been possible without Fran. Now, the three of us are going back to Chile. Javier, our beloved son, has recently joined us. In these few months he has been with us, Javier has fulfilled our lives with joy and happiness, bringing light whenever I needed it to finish this thesis.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

The study of routing problems has given rise to major developments in the fields of Operations Research (OR). In particular, the Vehicle Routing Problem (VRP) has motivated the development of many exact algorithms and heuristics. In the VRP, a planner designs minimum-cost-delivery routes from a depot to a set of geographically distributed customers, subject to capacity and business constraints. This problem is an important component of distribution systems and, in practice, several variants of the problem that exist are motivated by the diversity of operations rules and constraints in real-life applications. The VRP generalizes the Traveling Salesman Problem (TSP), so any of its variants present a computational challenge. We study a stochastic variant of the VRP and the Inventory Routing Problem (IRP), a problem that was initially studied as a variant of the VRP.

In Chapter 2, we study the Vehicle Routing Problem with Probabilistic Customers (VRP-PC), a two-stage stochastic optimization problem that is a fundamental building block within the broad family of stochastic routing models. In the first stage, a dispatcher determines a set of vehicle routes serving all potential customer locations, before actual requests for service realize. In the second stage, vehicles are dispatched after the subset of customers requiring service is observed; a customer not requiring service is skipped from its planned route at execution. The objective is to minimize the expected vehicle travel cost by assuming known customer realization probabilities. We propose a column generation framework to solve the VRP-PC to a given optimality tolerance. Specifically, we present two novel algorithms, one that underapproximates a solution's expected cost, and another that uses its exact expected cost. Each algorithm is equipped with a route pricing mechanism that iteratively improves the approximation precision of a route's reduced cost; this produces fast route insertions at the start of the algorithm and reaches termination conditions at the end

of the execution. Compared to branch-and-cut algorithms for the VRP-PC using arc-based formulations, our framework can more readily incorporate sequence-dependent constraints such as customer time windows. We provide *a priori* and *a posteriori* performance guarantees for these algorithms, and demonstrate their effectiveness via a computational study on instances with realization probabilities for customers ranging from 0.5 to 0.9.

In Chapter 3, we consider a variant of the Inventory Routing Problem (IRP), the Continuous Time IRP (CIRP). In time dependent models, such as the CIRP, the objective is to find the optimal times (continuous) at which activities occur and resources are utilized. These models arise whenever a schedule of activities needs to be constructed. A common approach consists of discretizing the planning time and then restricting the decisions to those time points. However, this approach leads to very large formulations that are intractable in practice. In the CIRP, a company manages the inventory of its customers, resupplying a single product from a single facility during a finite time horizon. The product is consumed at a constant rate (product per unit of time) by each customer. The customers have local storage capacity. The goal is to find the minimum cost delivery plan that ensures that none of the customers run out of product during the planning period. We investigate time-expanded network formulations that can form the basis of a Dynamic Discretization Discovery (DDD) algorithm and demonstrate in an extensive computational study that they, by themselves, produces provably high-quality, often optimal, solutions.

In Chapter 4, we study the Continuous Time IRP with Out-and-Back Routes (CIRP-OB): a vehicle route starts at the depot, visits a single customer, and returns to the depot. We develop the full DDD algorithm to solve the CIRP-OB by using partially constructed time-expanded networks. This method iteratively discovers the time points needed in the network to find optimal solutions. We test this method on randomly generated instances with up to 30 customers, where provable optimal

solutions are found in most cases.

CHAPTER 1
INTRODUCTION AND BACKGROUND

The study of routing problems has given rise to major developments in the fields of Operations Research (OR). In particular, the Vehicle Routing Problem (VRP), which was introduced in [1] under the title "The Truck Dispatching Problem", has motivated the development of many exact algorithms and heuristics. In the VRP, a planner designs minimum-cost-delivery routes from a depot to a set of geographically distributed customers, subject to capacity and business constraints. This problem is an important component of distribution systems, and, in practice, several variants of the problem are motivated by the diversity of operations rules and constraints in real-life applications.

The VRP generalizes the Traveling Salesman Problem (TSP) but is much more difficult to solve in practice, so any of its variants present a computational challenge. In the stochastic VRP, not all the information is provided in the beginning, but it is revealed over time. Uncertainty may affect any of the input data. The most common cases are stochastic customer requests, where a customer needs to be served with a given probability; stochastic times, in which service time or travel times or both are modeled by random variables; and, stochastic customer demands.

Another problem that was initially studied as a variant of the VPR is the Inventory Routing Problem (IRP) [2]. The IRP integrates inventory management, vehicle routing, and delivery-scheduling decisions. It is aimed at reducing logistics costs by giving the replenishment decisions for products delivered to customers from one or multiple facilities (supply points). It benefits both stakeholders, in that the vendors can coordinate deliveries to customers while the buyers do not need to allocate efforts to inventory control.

There exists a vast literature on the VRP and on the IRP [3, 4, 5, 6]. Several books and survey articles can be found about these problems.

This thesis consists of my work with Alejandro Toriello and Mathias Klapp during the first year of my PhD program, and of my work with Natashia Boland and Martin Savelsbergh during my second, third and fourth years. In these two lines of work, we studied routing problems and exact algorithms, first for a stochastic variant of the VRP, and then for a continuous time version of the IRP.

1.1 Vehicle Routing Problem with Probabilistic Customers (VRP-PC)

We study the VRP-PC, a variant of the VRP where the subset of customers requiring service is random and follows a known probability distribution. As a recourse rule, a customer that does not require service is skipped in its corresponding route, while keeping the rest of the route's sequence. The objective is to minimize the expected cost of the delivery plan. This problem is in the class of a priori optimization problems. Among the motivations to study these problems are that

- information may be not available far enough in advance to create optimal delivery plans for customers that require a visit;
- regularity of service can be beneficial for both the customers and the drivers: the customers will be served at roughly the same time each day they require service, and the drivers can become very familiar with their routes; and,
- starting from an a priori tour can be useful as a starting point for reoptimization.

For more details about a priori routing optimization, see [7].

A Branch-and-Price algorithm can be used to solve the VRP-PC. The column generation methodology has been successfully applied to the VRP by numerous researchers. Often the master problem is simply stated as a set partitioning problem to which column generation is applied, where each element of the partition or column

is a route that serves a subset of customers. A direct application of this formulation is not practical because the number of potential routes is exponentially large. The algorithm starts with a subset of columns and additional columns are generated as needed by a pricing subproblem. An integer solution is found by a Branch-and-Bound algorithm.

1.2 Continuous Time Inventory Routing Problem (CIRP)

Time-dependent decision models are pervasive in applications since they occur whenever a schedule of activities needs to be constructed. Time-dependent decision models seek to find the optimal times (continuous) at which activities occur and resources are utilized. A common technique for modeling these kinds of problems is discretization. Time discretization leads to formulations on a time-expanded network, in which a node represents both a location and a time. However, a trade-off arises when time is discretized. The granularity of the discretization impacts the solution quality (better candidate solutions are identified), but a too fine discretization might produce a huge formulation that is intractable in practice. Another option is to consider continuous variables to model time and linearize nonlinear terms, resulting in "big-M" constraints. These kinds of formulations are more compact, but they are weak and therefore difficult to solve in practice.

In [8], the authors propose a new algorithm for solving time-dependent models: the Dynamic Discretization Discovery (DDD) algorithm. The algorithm produces an optimal continuous time solution without explicitly modeling each time point in time. This algorithm achieves the benefit of a fine discretization (high-quality solution) without ever constructing a huge formulation. Starting with a partially time-expanded network, the algorithm refines the time discretization, based on the analysis of the solution obtained at each iteration. Since this solution is a relaxation of the continuous time problem, an optimization problem is solved to check whether

the solution can be converted (or repaired) into a continuous time solution with the same cost (so the solution is optimal). If not, then time points that can be added to the network are identified to obtain a different solution in the next iteration. A flowchart of this algorithm can be found in Figure 1.1.

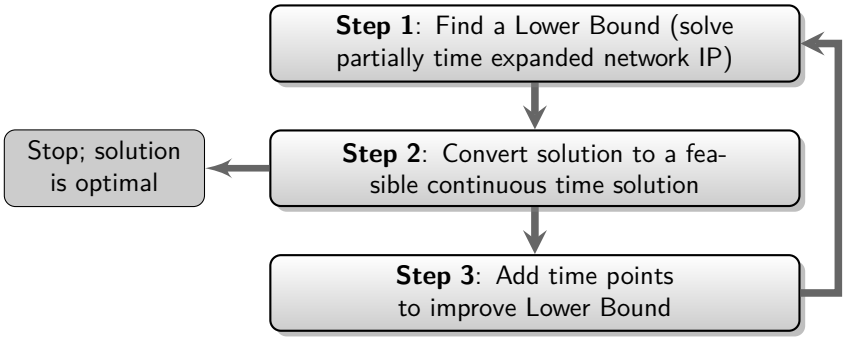

Figure 1.1: Flowchart of DDD algorithm.

The DDD algorithm is initially used for the service network design problem, but then it has been used for other problems such as time-dependent shortest path problems [9] and traveling salesman problems with time windows [10].

The variant we consider in this work is motivated by the IRP encountered by companies in the liquid gas industry. These companies produce liquefied gases, e.g., liquid oxygen, liquid nitrogen, or liquid argon, install tanks at their customers' premises, and guarantee minimum product availability at any time. Customers use (consume) product at a certain rate (which can differ at different times of the day) often 24 hours per day (e.g., liquid oxygen in hospitals.) Thus, the amount of product that can be delivered to the tank changes at the same rate. The companies continuously monitor product usage and tank inventory levels so that they can produce cost-effective delivery schedules that meet their service commitments (i.e., the guaranteed minimum

product availabilities). In practice, the companies tend to have customers that require multiple deliveries per day as well as customers that require as few as one or two deliveries per week. As a consequence, the use of a continuous time variant of the IRP is most appropriate in these settings in that it provides the most accurate representation of the system. Also relevant is the fact that the company contracts typically specify that the customers own/purchase the product upon delivery, which means that the companies do not have to consider product holding costs at the customer.

1.3 Overview

This thesis presents three additional chapters: chapters 2, 3 and 4. Each of them is either an accepted, under review or working paper, so each of them can be read independently.

In Chapter 2, we study the VRP-PC. We propose two novel algorithms to solve the VRP-PC, one that approximates the expected cost of the solution and another that uses the exact value. Each algorithm considers a pricing subproblem that identifies routes that are needed in the problem and provides optimal condition guarantees at the end of the execution. In addition, we provide *a priori* and *a posteriori* optimality gaps of the solution, and carry out a computational study on instances with realization probabilities ranging from 0.5 to 0.9.

In Chapter 3, we study the CIRP and investigate critical components of the DDD algorithm. We demonstrate in an extensive computational study that these components are sufficient to produce provable, high-quality, often optimal, solutions.

In Chapter 4, we consider a simpler version of the CIRP, the CIRP with out-and-back routes (CIRP-OB). In this problem, a vehicle route starts at the depot, visits a single customer, and returns to the depot. The DDD algorithm is fully developed and implemented and finds optimal solutions for instances with up to 30 customers.

CHAPTER 2
BRANCH-AND-PRICE FOR ROUTING WITH PROBABILISTIC CUSTOMERS

2.1 Introduction

The Vehicle Routing Problem (VRP) and its variants are widely studied within the transportation and operations research communities, and used in a variety of applications in freight and urban transportation and logistics, as well as in other areas [11, 12, 13]. Stochastic VRPs are extensions of their deterministic counterparts where some instance parameters are unknown while planning and/or executing a solution, and decisions must account for this uncertainty. *A priori* VRPs are particular versions of stochastic VRPs modeled as two-stage stochastic optimization problems; see [7] for a survey. In the first stage, some of the parameters are random variables and the decision maker plans an initial solution. In the second stage, the planned solution is executed after the realization of the random parameters. Typically, a simple recourse rule is used to modify the initial plan during its execution. A desired feature in the first stage solution is to proactively account for parameter uncertainty and the second-stage recourse.

We introduce a set partition model to study the *a priori* VRP with probabilistic customers (VRP-PC). In the VRP-PC, the subset of customers requiring service is random and follows a known probability model. In the first stage, the decision maker plans a set of vehicle routes dispatched from the depot and visiting all potential customer locations. Vehicles are dispatched in the second stage after observing the subset of customers requiring service. The second-stage recourse rule modifies first-stage routes by skipping locations without a service requirement, maintaining the

established sequence of each route for the visited customers. The objective is to minimize the expected vehicle travel cost, accounting for this recourse rule. The VRP-PC extends the Probabilistic Traveling Salesman Problem P-TSP [14, 15], a single-vehicle version of the problem. Our formulation is the first route-based model for the VRP-PC, and contrasts with the arc-based VRP-PC formulation originally proposed in [16].

2.1.1 Applications of VRP-PC

There are many applications of the VRP-PC. As an extension of the P-TSP, it may be used in multi-vehicle and constrained problems related to the P-TSP. It is a natural model when the set of customers requiring service is unknown in the planning stage, but it is infeasible or impractical to optimize routes at execution, *e.g.*, [17]; this might happen when the decision maker does not have enough computational resources to optimize routes in real time [18], or when maintaining the initial visit sequence is desirable or required, *i.e.*, because of gains in operational efficiency produced by drivers executing the same plan every day, or because customers expect to meet the same driver at roughly the same time when they require service.

Some application examples include technician routing and scheduling, and cash-in-transit vehicle routing problems (CTVRP). In the technician routing and scheduling problem there is a limited crew of technicians serving requests requiring specialized tools and skills; such a problem arises in public services, telecommunication services and equipment maintenance operations. In such settings, routes that maintain visit sequences make it easier to meet technical requirements; see *e.g.*, [19, 20, 21]. In the CTVRP [22], vehicles are assumed to transport banknotes, coins and other items of high value. Vehicle routes are therefore vulnerable to robberies, and a carefully planned route is desirable to reduce security risks. A priori route planning maintains a daily routine and improves aspects of customer security, such as time window

constraints and visit order.

A novel application involves using the VRP-PC as a building block to solve more complex dynamic and stochastic VRPs; see *e.g.*, [23, 24, 25]. In many dynamic settings, the model can be leveraged to design heuristic dynamic policies by maintaining and periodically updating a feasible route plan via a rollout procedure [26, 27, 28, 29]. For example, our particular motivation for studying the VRP-PC stems from urban distribution problems in same-day delivery [30, 31]. In such a setting, customer delivery requests realize dynamically over the operating period as other previously known orders are being prepared, loaded into vehicles and dispatched from a depot to customer locations in vehicle routes. Therefore, an effective dispatch policy requires adapting and reacting to newly revealed information and involves online reoptimization of planned routing decisions. One type of high-quality heuristic policy carries a plan serving a subset of known and pending delivery requests, along with a set of potential customer locations (*e.g.*, neighborhoods, city blocks) to account for future expected routing costs. At the time of dispatch, each route in fact only visits a location if an order realizes there before the vehicle route begins. An *a priori* solution of this kind is not by itself necessarily desirable, as it could result in significant wasted time for the vehicle if executed when many orders do not realize. Nevertheless, one can "roll out" such an *a priori* solution, enforcing in the model that such a route only includes already realized orders; the remainder of the plan is a proxy for the expected duration, length and/or cost of subsequent "average" routes in an "average" day.

In some circumstances, same-day delivery systems may dynamically choose whether to accept customer requests or not. Our model can be used to guide order acceptance decisions while proactively considering potential future orders [32].

2.1.2 Contribution

We introduce a set covering model for the VRP-PC and propose a novel exact approach for it, which is compatible with a wide variety of route-dependent constraints, including useful sequence-dependent constraints, such as time windows and precedence constraints. An exact algorithm for the P-TSP is proposed in [33], which is a specialized implementation of the integer L-shape method [34]. This method formulates the P-TSP as a two-stage integer linear program with edge-based variables, and replaces the expected route cost in the objective by a variable $\theta \geq 0$. As integer routes are found, the real expected cost of these solutions is evaluated to dynamically generate "optimality cuts" on θ and correct its value.

The integer L-shape method operates on a formulation with edge variables. However, many practical sequence-dependent constraints are difficult to model with edge variables, and often have weak relaxations. In same-day delivery problems, pertinent examples of such constraints include delivery deadlines and order release times.

Therefore, to our knowledge the work on routing with probabilistic customers considering hard sequence-dependent constraints has been restricted to heuristics; see [35, 36]. Nonetheless, in deterministic problems, route-based formulations solved via column generation and branch-and-price (B&P) are arguably more effective to optimize routing models with such constraints (see *e.g.*, [37]). We address this gap in the literature and propose a B&P framework to solve the VRP-PC. In particular, our contributions are:

1. We present two different and independent column generation algorithms, one that underestimates a feasible solution's cost and another that uses its exact expected cost. Both algorithms use route generation subroutines that compute estimates of a route's reduced cost. The algorithms' pricing problems iteratively increase the precision of a route's expected cost estimate, yielding fast route

generation at the start while still reaching termination conditions at the end of the execution.

2. We provide *a priori* and *a posteriori* performance guarantees for these algorithms, which allow us to determine solution quality before and after a lower bound on the optimal value is computed.

3. We implement a prototypical branch-and-price scheme for the VRP-PC and conduct a computational study, concluding that both algorithms find good solutions using only a few number of precision updates. The algorithm's empirical convergence takes few iterations, and depends on the customer realization probabilities.

After closing this section with a literature review, the remainder of the article is organized as follows. Section 2 reviews the B&P approach for the VRP and formulates the VRP-PC. In Section 3 we present two column generation algorithms for the VRP-PC, discuss incorporating them into B&P, and give convergence guarantees and approximate optimality conditions. Section 4 presents a computational study on modified Solomon instances [38], and Section 5 concludes.

2.1.3 Literature Review

A priori optimization is routinely applied in stochastic routing problems [39, 40, 41]. Different sets of uncertain parameters within VRPs are modeled through different *a priori* optimization problems. For example, the VRP with stochastic travel times (VRP-STT) refers to uncertainty in travel times between locations [42, 43] and the VRP with stochastic demand (VRP-SD) [44, 17] refers to a VRP where the customer demand is a random variable realized at the moment of service.

The *a priori* VRP with probabilistic customers (VRP-PC) is a stochastic VRP where the subset of customers requiring service is random and follows a known proba-

bility distribution. The seminal work in [14, 15] formally introduces the Probabilistic Traveling Salesman (P-TSP) and develops a closed-form expression for a route's expected cost in the case of homogeneous probabilities. [45] extends the P-TSP to heterogeneous probability distributions, and additional results include [46, 17, 39, 40].

In [34], the authors present the integer L-shape method to exactly solve *a priori* optimization problems; the algorithm gradually improves an estimate of the expected cost via dynamic generation of cutting planes in an integer program. In [33], the integer L-shape method is used to solve an edge-based formulation of the P-TSP. This method is also implemented in [16] to design an exact algorithm for a VRP with both probabilistic customers and stochastic demand.

Truncation approaches have been proposed by [47] for the P-TSP; they underestimate the true cost by computing only some terms of the expected cost of a route. In [48], the authors underestimate the true cost using an approximation function; this function balances the amount of speedup for solving algorithms and the quality of the approximation. Such approximation functions have been used to speed up heuristics [49].

For deterministic routing, B&P approaches have been successfully applied to solve many VRP variants to optimality, and readily handle sequence dependent constraints. These approaches formulate the VRP as a set covering route-based formulation; see *e.g.*, [50, 51, 52] and references therein. Recently, B&P has been adapted to solve the VRP-SD [53, 54, 55] and the VRP-STT [56, 57]. We are not aware of a B&P scheme for the VRP-PC, and the technical hurdles needed to overcome probabilistic customers in B&P are in several respects different from these previous works.

The route pricing problem within a VRP set partitioning formulation is the Elementary Shortest Path Problem with Resource Constraints (ESPPRC). Because of the ESPPRC's theoretical complexity and empirical difficulty, a relaxed version called

the Shortest Path Problem with Resource Constraints (SPPRC) [37] is usually solved instead; the SPPRC relaxes the elementary condition to allow visiting a customer more than once. The SPPRC with k-cycle elimination [58] is an intermediate relaxation that only allows paths with cycles having at least $k+1$ nodes.

2.2 Model Formulation and Preliminaries

The deterministic VRP is a starting point to study the more complex probabilistic variant. We start by briefly describing a deterministic VRP set partitioning model and its classic column generation framework, including a relaxation technique for its route pricing subproblem; these concepts are widely studied in the VRP and column generation literature, *e.g.*, [37, 13]. Later, we introduce the VRP with probabilistic customers and discuss how to fit the previous column generation framework in this two-stage stochastic problem. We also show how chance constraints, a useful modeling tool in stochastic optimization, can be incorporated in the model.

2.2.1 Deterministic VRP

The deterministic VRP entails designing a set of vehicle routes, each starting and ending at a depot, that feasibly serve a set $C = \{1, ..., n\}$ of customers at minimum total travel cost. Each customer $i \in C$ has demand $d_i > 0$ that must be served by one route; that is, we do not allow split deliveries. We consider homogeneous vehicles with capacity $q \geq \max_i d_i$, so that the total demand served on any route does not exceed q. A vehicle can only perform one route, and we may have a limit on the total number of routes. We also assume no fixed route setup costs, though these may be included without substantially changing the model.

Define the set $N := C \cup \{0, n+1\}$ of nodes having all customers plus 0 and $n+1$, both of which represent the depot. A vehicle route departs node 0, visits a subset of customer nodes, and ends at node $n+1$. Traveling between any two nodes

$i, j \in N$ costs $c_{ij} \geq 0$ and takes $t_{ij} \geq 0$ time, and we assume both parameter sets satisfy the triangle inequality. When $i \in C$, the value t_{ij} may include service time at i, and we may also have a route duration limit T, with $T \geq t_{ij}$ for all arcs. For the sake of exposition, we describe only vehicle capacity and route duration as limiting resources, but other complex and sequence-dependent constraints, such as service time-windows, order release dates and customer precedence constraints amenable to column generation and B&P may be handled similarly. We only require that all applicable constraint parameters are integers, possibly by re-scaling.

2.2.2 Column Generation and Branch-and-Price

To specify the column generation framework, define R_n as the set of all feasible vehicles routes; each route $r \in R_n$ corresponds to an elementary path of nodes in C starting at 0, ending at $n+1$ and satisfying any required constraint. Let c_r be the cost of route r and $\alpha_r^i \in \{0, 1\}$ be the number of times $i \in C$ is visited by r.

Define the binary variable y_r, equal to 1 if route r is selected and 0 otherwise. The VRP can then be formulated as a set partitioning integer linear problem, where its linear relaxation is given by

$$\min_{y \geq 0} \quad \sum_{r \in R_n} c_r y_r \tag{2.1a}$$

$$\text{s.t.} \quad \sum_{r \in R_n} \alpha_r^i y_r = 1, \qquad i \in C. \tag{2.1b}$$

The number of routes in R_n can be exponentially large as a function of n, making it difficult to solve (2.1) explicitly with an LP solver. Instead, routes can be generated dynamically using column generation. To solve the VRP set partition LP relaxation, we use an algorithm with two main components. Initially, we solve model (2.1) only considering a small subset $\tilde{R} \subset R_n$ of feasible routes. The optimal dual solution of this restricted LP is then used to identify profitable columns in $R_n \setminus \tilde{R}$ via a pricing

subproblem. We add these new columns to $\tilde{R}$, then execute a new run of the LP solver. The procedure is repeated until no more profitable columns are found in $R_n \setminus \tilde{R}$.

Let ρ_i for $i \in C$ be the dual LP variable related to constraint (2.1b). The pricing problem to generate routes based on the current restricted LP solution is

$$\min_{r \in R_n} \left\{ c_r - \sum_{i \in C} \alpha_r^i \rho_i \right\}. \tag{2.2}$$

A non-negative optimal value of (2.2) certifies optimality for the restricted version of (2.1). Conversely, a negative value indicates the existence of a profitable route $r \in R_n \setminus \tilde{R}$.

Problem (2.2) is known as the Elementary Shortest Path Problem with Resource Constraints (ESPPRC), and it is strongly NP-hard [59]. The Shortest Path Problem with Resource Constraints (SPPRC) is a relaxation of (2.2) that allows multiple visits to a customer; this relaxed version can be solved by dynamic programming (DP) in polynomial time as a function of n and the resources' parameters. In the case of vehicle capacity, the applicable parameter is vehicle capacity q; similarly, for time-based constraints the corresponding parameter is the duration limit T. A relaxed route r can have cycles, but cannot make more than n customer visits, and may in fact be further constrained, since resources are increasingly consumed along any path. We consider k-cycle elimination for the SPPRC (SPPRC-k) [58] to improve relaxation quality; in this setting any path with cycles having fewer than $k+1$ nodes is not allowed. Let R_k be the set of relaxed routes with k-cycle elimination, and observe that since elementary paths exclude all cycles up to length n, this definition is consistent with our use of R_n.

The SPPRC is defined over a network of partial route states $G = (V, A)$. Each state $v \in V$ is specified by a tuple $v = (S_v, d_v, t_v)$, where S_v is a sequence of visited

locations in the partial route, d_v is the vehicle capacity already used by the partial route, and t_v is the partial route's end time. When the problem has other route resources, these are tracked in a similar fashion. The ESPPRC carries the complete node visiting sequence from the depot in S_v, while the SPPRC-k relaxation only records the latest k visited nodes in the sequence. The transition cost between states v and u is defined by $c_{\ell(S_v),\ell(S_u)} - \rho_{\ell(S_v)}$, where $\ell(S)$ is the last node in sequence S.

The DP optimality equations for the SPPRC-k are

$$F([0], 0, 0) = 0, \tag{2.3a}$$

$$F(S, d, t) = \min_{(i,j)\in N^2} \Big\{ F(\bar{S}, \bar{d}, \bar{t}) + (c_{ij} - \rho_i) : (i, j) = (\ell(\bar{S}), \ell(S)), i \neq n+1, f_{ij}(\bar{d}, \bar{t}) \leq (d, t) \Big\}, \tag{2.3b}$$

where $\rho_0 = 0$. The value of $F(S, d, t)$ is the smallest possible reduced cost for a partial route starting from the depot, ending with sequence S (where $|S| \leq k$), having consumed resources (d, t). The relaxed route with minimum reduced cost is given by the minimum over all values $F(S, q, T)$ with $\ell(S) = n + 1$. In the recursion, $\bar{S}$ is a sequence with $|\bar{S}| \leq k$ that can be extended to S; that is, either $|\bar{S}| < k$ and appending $\ell(S) = j$ to it yields S, or $|\bar{S}| = k$ and S consists of deleting the first element and appending j. The function f_{ij} is called a resource extension function [37] and models resource consumption, *i.e.*,

$$f_{ij}(d, t) = (d + d_j, t + t_{ij}).$$

The function can be defined more generally and include any other resource consumed along a route. We consider the Bellman-Ford labelling algorithm to solve (2.3); the number of states satisfies $|V| = O(n^k qT)$, and therefore the running time of the algorithm is $O(n^{k+1}qT)$.

The optimal values of (2.1) may be fractional; however, branching on the y variables is generally impossible (as the pricing problem cannot be readily updated) and would lead to extremely unbalanced search trees. Instead, for VRP the typical B&P scheme [37] branches on the implied arc variables x_{ij} that indicate whether any route in the solution travels directly from i to j. Adjusting the pricing SPPRC-k model for these branching decisions simply involves forcing or forbidding some actions at certain states. In our implementation, we also initially branch on the number of routes in the solution, as originally proposed in [51].

2.2.3 VRP with Probabilistic Customers

We now consider a VRP with probabilistic customers (VRP-PC), where an initial solution covering all customers C is planned, but only a subset of customers will actually require service. As a recourse rule, a customer that does not require service is skipped in its corresponding route, while keeping the rest of the route's sequence; for many VRP constraints of interest (such as vehicle capacity and route duration), this ensures that if the planned route is feasible when all customers are serviced, it remains feasible for any realized subset of customers in the probabilistic context. We assume a known, independent customer realization probability $p_i \in (0, 1]$ for each $i \in C$ and use $p_0 = p_{n+1} = 1$. The objective is to minimize the expected vehicle travel cost. The problem's stochasticity only affects expected route costs, and therefore any constraints are managed identically to the deterministic VRP.

For a route $r \in R_n$, let $n_r \leq n$ be the number of planned customer visits, and let $r(i) \in N$ represent the i-th planned visit in r (with $r(0) = 0$, $r(n_r + 1) = n + 1$). The

expected cost $\mathbb{E}(c_r)$ of the route is

$$\begin{aligned}\mathbb{E}(c_r) &= \sum_{i=0}^{n_r}\sum_{j=i+1}^{n_r+1}\left(p_{r(i)}p_{r(j)}\prod_{\ell=i+1}^{j-1}(1-p_{r(\ell)})\right)c_{r(i),r(j)} \\ &= \sum_{\ell=1}^{n_r}\sum_{i=0}^{n_r-\ell+1}\left(p_{r(i)}p_{r(i+\ell)}\prod_{j=i+1}^{i+\ell-1}(1-p_{r(j)})\right)c_{r(i),r(i+\ell)} =: \sum_{\ell=1}^{n_r} H_r^\ell. \end{aligned} \tag{2.4}$$

This expected cost can be computed as a sum of n_r nonnegative terms as in (2.4), where each H_r^ℓ includes the expected costs corresponding to arcs that skip exactly $\ell - 1$ customers. For example, the term $H_r^1 = \sum_{i \le n_r} p_{r(i)}p_{r(i+1)}c_{r(i),r(i+1)}$ considers all arcs between consecutive customers. The term H_r^2 includes all arcs that skip one customer in the sequence, and so on. Since they are all non-negative, a summation of any subset of the H_r^ℓ terms provides a lower bound for $\mathbb{E}(c_r)$. In particular, this includes the possible case in which all the customers in a route do no request service, $H_r^{n_r+1}$, and the cost of that realization is zero, $H_r^{n_r+1} = 0$.

A set partitioning relaxation for the VRP-PC is

$$f(R_k) := \min_{y \ge 0}\left\{\sum_{r \in R_k}\mathbb{E}(c_r)y_r : \sum_{r \in R_k}\alpha_r^i y_r = 1, i \in C\right\}. \tag{2.5}$$

Compared to a deterministic VRP relaxation, the feasible region remains unaltered, but the objective function considers each route's expected cost. As in the deterministic case, we can relax the set of feasible routes from R_n to a larger set R_k that only excludes k-cycles; we make the dependence on the set of feasible routes explicit and use $f(R_k)$ to denote the optimal value of this relaxation as a function of the considered route set. In this case, the definition of $\mathbb{E}(c_r)$ can be extended to include routes with repeated customer visits, by simply setting $c_{r(i),r(j)} = 0$ when $r(i) = r(j)$ and keeping all other values the same. In other words, this definition treats repeated visits to customer i in a route as independent copies of it, each of which requires service independently with probability p_i. As in the deterministic problem, such a defini-

tion guarantees that routes with repeated visits never appear in an optimal integer solution.

2.2.4 Chance Constraints

As stated, our model handles resource consumption deterministically. For example, we construct routes that satisfy vehicle capacity even if every customer realizes. In our motivating applications [30, 31] this is indeed necessary. Furthermore, if realization probabilities are high, say $p_i \geq 0.9$ for all customers i, it may be desirable to guarantee the route's feasibility under any circumstance. Therefore, most of our exposition and the instances in our computational study use this approach.

However, in other cases, guaranteeing route feasibility with absolute certainty may result in overly conservative routes with significant amounts of wasted capacity. A systematic approach to this issue involves replacing a deterministic constraint with a chance constraint, which for example stipulates that the probability of realized customers' demand exceeding capacity should be small. By incorporating a chance constraint, we keep the same two-stage planning method used in the deterministic case, but we allow the planned routes to possibly violate resource capacity, as long as the violation is unlikely. This technique has been explored before for the VRP-SD [54]; we suggest one possible way to include it here.

Let D_i be a random variable representing the realized demand at customer i; so D_i follows a scaled Bernoulli distribution with $\mathbb{P}(D_i = d_i) = p_i$ and $\mathbb{P}(D_i = 0) = 1 - p_i$. For a planned route r, we could replace the deterministic vehicle capacity constraint $\sum_{i \leq n_r} d_{r(i)} \leq q$ with the chance constraint

$$\mathbb{P}\left(\sum_{i=1}^{n_r} D_{r(i)} > q\right) \leq \eta,$$

for an appropriately chosen tolerance $\eta > 0$; for example, $\eta = 0.1$ means the vehicle's

capacity can satisfy realized demand with at least 90% probability.

The issue is how to model such a chance constraint so that it is amenable to pricing via a recursion similar to (2.3). Applying a Chernoff bound, we obtain

$$\mathbb{P}\left(\sum_{i=1}^{n_r} D_{r(i)} > q\right) \leq e^{-q\tau} \prod_{i=1}^{n_r} \mathbb{E}\left[e^{D_{r(i)}\tau}\right] = e^{-q\tau} \prod_{i=1}^{n_r} \left(1 - p_{r(i)} + p_{r(i)} e^{d_{r(i)}\tau}\right), \quad \forall\ \tau > 0,$$

where we use the fact that the D_i's are independent, and $\tau > 0$ can be chosen based on problem parameters. If the quantity on the right does not exceed the tolerance η, we guarantee that the route is feasible for the chance constraint. In other words, the route is feasible if for some $\tau > 0$ we have

$$e^{-q\tau} \prod_{i=1}^{n_r} \left(1 - p_{r(i)} + p_{r(i)} e^{d_{r(i)}\tau}\right) \leq \eta,$$

or equivalently,

$$\frac{1}{\tau} \sum_{i=1}^{n_r} \ln\left(1 - p_{r(i)} + p_{r(i)} e^{d_{r(i)}\tau}\right) \leq q + \frac{\ln \eta}{\tau}.$$

Note that as $\tau \to \infty$, this constraint recovers the deterministic counterpart. By defining a new "capacity" $\hat{q} := q + (\ln \eta)/\tau$ and new "demands" $\hat{d}_i := (1/\tau) \ln\left(1 - p_{r(i)} + p_{r(i)} e^{d_{r(i)}\tau}\right)$, we can incorporate the chance constraint into the pricing recursion (2.3).

As an example, suppose a planned route has ten customers, each customer i with $d_i = 2$ and $p_i = 0.5$. Choosing $\tau = 1$, this gives a probabilistic "demand" parameter of $\hat{d}_i \approx 1.43$. To be feasible in the deterministic version of the vehicle capacity constraint, we would require a capacity of 20. If we use $\eta = 0.1$, we obtain via the bound that capacity can be as low as $10 \cdot \hat{d}_i - \ln 0.1 \approx 16.64$ and the route would remain feasible for the chance constraint. This is still an approximation; we can verify by direct calculation that even with a capacity of 14 the route would still be feasible for the chance constraint.

2.3 Column Generation for VRP-PC

We next study how to price columns for the VRP-PC by approximating a route's expected reduced cost. Later, we provide two solution algorithms based on the VRP-PC column generation model introduced in Section 2.2.2.

The pricing problem for the VRP-PC master problem in (2.5) with k-cycle elimination considers the expected cost of each relaxed route $r \in R_k$ and is defined by

$$\min_{r \in R_k} \left\{ \mathbb{E}(c_r) - \sum_{i \in C} \alpha_r^i \rho_i \right\}, \tag{2.6}$$

where the ρ_i are again dual multipliers. Even though we relax the route set to R_k, the expected cost formula (2.4) depends on the entire customer visit sequence and increases the subproblem's difficulty. To deal with this additional difficulty, we further relax the expected cost expression and only consider the first k terms,

$$E^k(c_r) := \sum_{\ell=1}^{k} H_r^\ell,$$

where we include arcs that skip up to $k-1$ customers in the sequence. $E^k(c_r)$ is a valid lower bound for $\mathbb{E}(c_r)$ for $k \in \{1, \ldots, n_r\}$, as we show here.

Proposition 1. *For any route $r \in R_1$ and $k \in \{1, \ldots, n_r\}$, the k-term approximation of the expected cost $\mathbb{E}(c_r)$ is monotone non-decreasing in k, i.e., $E^k(c_r) \leq E^{k+1}(c_r)$. Moreover, if $\bar{c}_r$ is the deterministic cost of visiting all customers in r, $E^{n_r}(c_r) = \mathbb{E}(c_r) \leq \bar{c}_r$.*

Proof. $E^{k+1}(c_r) - E^k(c_r) = H_r^{k+1} \geq 0$, so this approximation is monotone non-decreasing. The exact expected cost considers all non-negative terms H_r^ℓ and is thus an upper bound to $E^k(c_r)$. Finally, the deterministic cost is no smaller than the expected value because under the triangle inequality, the cost of each possible realization is always less than or equal to the cost of visiting all customers. □

In the following sections, we provide two algorithms that exploit this lower bound $E^k(c_r)$ on the expected cost of a route r.

2.3.1 Updating Cost Algorithm

In our first algorithm, which we call the Updating Cost Algorithm (UCA), we obtain a lower bound for the optimal reduced cost of the relaxed master problem (2.5) by approximately solving subproblem (2.6) using a non-elementary path relaxation with k-cycle elimination (SPPRC-k), and replacing the exact expected cost $\mathbb{E}(c_r)$ with the approximation $E^k(c_r)$. When solving the SPPRC-k, we can adapt the DP recursion (2.3) to include expected arc costs corresponding to arcs that skip at most $k-1$ customers, thus allowing us to optimize with respect to $E^k(c_r)$.

We first address how well E^k approximates the true expected cost. Let $\hat{p} := \max_{i \in C} p_i$, $\check{p} := \min_{i \in C} p_i$, and let

$$\hat{c} := \max\Big\{\max_{i,j \in C} c_{ij}, \max_{i \in C}\Big\{\max\{c_{0i}, c_{i,n+1}\}/p_i\Big\}\Big\}.$$

This last quantity represents the most expensive arc cost in the graph, where we weigh arcs incident to the depot more heavily.

Lemma 1. *Let $y \geq 0$ satisfy $\sum_{r \in R_1} \alpha_r^i y_r = 1$ for each $i \in C$, and let $k \in \{1, \ldots, n\}$. Then*

$$\sum_{r \in R_1} y_r E^k(c_r) \leq \sum_{r \in R_1} y_r \mathbb{E}(c_r) \leq \sum_{r \in R_1} y_r E^k(c_r) + \delta_k,$$

where

$$\delta_k := \hat{c}\hat{p}^2 \sum_{\ell=k+1}^{n} (n - \ell + 2)(1 - \check{p})^{\ell-1}. \tag{2.7}$$

Although we state the result in terms of the largest route set R_1, the same guar-

antee applies to any smaller set R_k by taking $y_r = 0$ for $r \notin R_k$.

Proof. The first inequality is a consequence of Proposition 1. To prove the second, we first note that for any route $r \in R_1$,

$$\mathbb{E}(c_r) - E^k(c_r) = \sum_{\ell=k+1}^{n_r} H_r^\ell \leq \hat{c}\hat{p}^2 \sum_{\ell=k+1}^{n_r} (n_r - \ell + 2)(1-\check{p})^{\ell-1}.$$

Summing over the y_r values, we obtain

$$\begin{aligned}\sum_{r\in R_1} y_r(\mathbb{E}(c_r) - E^k(c_r)) &\leq \hat{c}\hat{p}^2 \sum_{r\in R_1} y_r \sum_{\ell=k+1}^{n_r} (n_r - \ell + 2)(1-\check{p})^{\ell-1} \\ &\leq \hat{c}\hat{p}^2 \sum_{\ell=k+1}^{n} (1-\check{p})^{\ell-1} \sum_{r\in R_1} y_r(n_r - \ell + 2) \leq \delta_k,\end{aligned}$$

where the last inequality follows from $\sum_r n_r y_r = \sum_r \sum_i \alpha_r^i y_r = n$ and $\sum_r y_r \geq 1$. □

This result allows us to gauge how closely we approximate our problem's true optimal cost if we use the approximate route cost E^k instead.

Theorem 2. *Suppose we replace $\mathbb{E}$ with E^k in (2.5), optimize with respect to this objective, and obtain solution y^k. Then*

$$\sum_{r\in R_k} y_r^k E^k(c_r) \leq f(R_k) \leq \sum_{r\in R_k} y_r^k \mathbb{E}(c_r) \leq \sum_{r\in R_k} y_r^k E^k(c_r) + \delta_k.$$

The analogous guarantee holds for the integral case: Suppose R^* is an optimal set of routes for the VRP-PC, and suppose $\bar{R}^k$ is an optimal set with respect to the approximate objective E^k. Then

$$\sum_{r\in \bar{R}^k} E^k(c_r) \leq \sum_{r\in R^*} \mathbb{E}(c_r) \leq \sum_{r\in \bar{R}^k} \mathbb{E}(c_r) \leq \sum_{r\in \bar{R}^k} E^k(c_r) + \delta_k.$$

Proof. The first inequality is a consequence of Lemma 1 and y^k's optimality with respect to E^k. The second follows from y^k's feasibility for (2.5), and the last from

Lemma 1. The same argument can be repeated for the second set of inequalities, restricting the analysis to integer y solutions. □

The theorem implies that if we approximately optimize (2.5) over routes R_k with objective E^k, we have the *a priori* guarantee that the solution we obtain will be within an additive gap of δ_k from $f(R_k)$. Similarly, if we embed this approximate optimization within a B&P algorithm, we are guaranteed to obtain an integer solution within δ_k of the optimal expected cost. In addition, after carrying out the optimization, we can calculate a tighter *a posteriori* gap by taking the difference of the solution's true expected cost with $\mathbb{E}$, minus its approximate expected cost as measured by E^k.

Corollary 3. *To achieve any desired additive optimality gap $\epsilon \geq 0$ in (2.5) (and in the integral problem via B&P), it suffices to choose $k = O(\log(n/\epsilon))$.*

Proof. By definition of δ_k, we have

$$\delta_k \leq \hat{c}\hat{p}^2 n \sum_{\ell=k+1}^{n} (1-\check{p})^{\ell-1} \leq \frac{\hat{c}\hat{p}^2 n(1-\check{p})^k}{\check{p}}.$$

Therefore, to guarantee $\delta_k \leq \epsilon$, it suffices for k to satisfy

$$(1-\check{p})^{-k} \geq \frac{\hat{c}\hat{p}^2 n}{\epsilon\check{p}} \quad \Longleftrightarrow \quad k \ln\left(\frac{1}{1-\check{p}}\right) \geq \ln\left(\frac{\hat{c}\hat{p}^2 n}{\epsilon\check{p}}\right). \qquad \square$$

Although this last result shows a logarithmic dependence on n, in practice we find that the δ_k values decrease quite rapidly, so that a small k suffices to provide a very tight gap. Table 2.1 provides an example of $\delta_k/\hat{c}$ values for $n = 50$ as a function of k and a uniform customer probability. Our computational results detailed in the next section verify this convergence and also explore the *a posteriori* gap in empirical terms.

Table 2.1: Sample $\delta_k/\hat{c}$ values as a function of k and $p = \check{p} = \hat{p}$, for $n = 50$.

$p \backslash k$	1	2	3	4	5
0.5	12.25	6.00	2.9375	1.4375	0.703125
0.7	10.41	3.06	0.8991	0.26406	0.077517
0.9	4.49	0.44	0.0431	0.00422	0.000413

```
1  input: integers K_0 ≤ K, initial set of routes R̃ and
     associated approximate costs E^{K_0}(c_r) for all r ∈ R̃;
2  set k ← K_0;
3  while k ≤ K do
4      solve restricted master problem using routes R̃
         and costs E^k;
5      solve pricing subproblem as SPPRC-k, obtain new
         route r;
6      if E^k(c_r) − Σ_{i∈C} α_r^i ρ_i < 0 then
7          include new column in master problem,
             R̃ ← R̃ ∪ {r};
8      else
9          k ← k + 1;
10         recompute approximate expected costs as
             E^k(c_r) for r ∈ R̃;
11     end
12 end
```

Algorithm 1: Column generation algorithm UCA.

Algorithm (1) details our implementation of this approximate optimization. Instead of starting from the desired approximation precision, the algorithm solves the column generation algorithm sequentially, with increasing precision at every step of the outer loop. Intuitively, we expect pricing subproblems with small k to be easy,

and thus to quickly find useful columns in the first steps; for larger values of k, we then obtain a few remaining columns that marginally improve the expected cost. Because the algorithm updates the cost approximation as it progresses, we name it the Updating Cost Algorithm (UCA).

UCA takes as argument an initial set of feasible routes $\tilde{R}$ and two positive integers $K_0 \leq K$; K_0 is the initial value for each route's expected cost approximation and K is the final value used in the approximation for column generation. In each step $k \in \{K_0, \ldots, K\}$, the algorithm searches for routes $r \in R_k$, approximating r's expected cost using $E^k(c_r)$. If the resulting route r has negative reduced cost, we include it in the set $\tilde{R}$ for the master problem; otherwise, we increase k and recompute the cost of the routes considered in the master using E^k.

UCA begins by generating routes in R_{K_0}, meaning some of these routes may in fact have cycles shorter than or equal to K. Therefore, we cannot claim that the algorithm optimizes the LP relaxation (2.5) over R_K. We can, however, make a weaker assertion.

Proposition 4. *The value returned by UCA with parameter K is a lower bound for $f(R_K)$.*

Proof. In its final iteration ($k = K$), the algorithm ensures that every route in R_K has non-negative reduced cost with respect to E^K. However, the algorithm can generate routes in its previous iterations, some of which could have cycles of length K or shorter. So UCA produces a solution that is optimal for a superset of R_K, where the inclusion may be strict. □

Finally, we verify that employing UCA within a B&P framework preserves the gap guarantee.

Corollary 5. *Using UCA within a B&P algorithm yields an integer solution with expected cost within an additive gap δ_K of optimal.*

Proof. The proof follows from Theorem 2 and Proposition 4 by noting that if UCA returns an integer solution, this solution must be optimal with respect to E^K. □

We finish this subsection by noting that although UCA has both an *a priori* and an *a posteriori* guarantee, it cannot guarantee a solution with lower expected cost than the solution implied by the deterministic VRP on the same instance. Therefore, in practice we warm start the algorithm with the deterministic solution, which also helps us fathom nodes in the search tree.

2.3.2 Fixed Cost Algorithm

The motivation for UCA and the use of the cost approximation E^k is the difficulty of the exact pricing problem (2.6). However, once we generate a particular route r, we can efficiently check its exact expected cost and thus its exact reduced cost. This motivates a second algorithm to approximately solve (2.5) and the VRP-PC, in which we include routes in the restricted master problem with their exact expected costs; because this second algorithm always keeps routes' true expected cost (instead of updating an approximation), we call it the Fixed Cost Algorithm (FCA).

Algorithm 2 details FCA. We again use two parameters $K_0 \leq K$ and increase the precision of the pricing problem and the expected cost approximation E^k for $k \in \{K_0, \ldots, K\}$; however, in this case we only add routes if their exact reduced cost (measured with the exact expected cost) is negative. This has the benefit of including columns in the master with their exact objective value, but the disadvantage that we may have an inconclusive pricing outcome, where a route with negative approximate reduced cost in fact has non-negative reduced cost. Such an inconclusive outcome triggers an increase in the pricing precision until we reach K, at which point the algorithm terminates.

```
input: integers K_0 ≤ K, initial set of routes R̃ with
  expected costs 𝔼;
set k ← K_0;
while k ≤ K do
    solve restricted master problem with routes R̃ and
     costs 𝔼;
    solve pricing subproblem as SPPRC-k with
     approximate cost E^k, obtain new route r;
    if E^k(c_r) − Σ_{i∈C} α_r^i ρ_i ≥ 0 then
        terminate;
    else if 𝔼(c_r) − Σ_{i∈C} α_r^i ρ_i < 0 then
        include new column in master problem,
         R̃ ← R̃ ∪ {r};
    else
        k ← k + 1;
    end
end
```

Algorithm 2: Column generation algorithm FCA.

Theorem 6. *The non-basic routes in the solution produced by algorithm FCA have reduced cost bounded below by* $-\delta_K$.

Proof. If the algorithm terminates because the minimum approximate reduced cost is non-negative, the current solution is optimal. Therefore, assume the algorithm terminates because the final route $\tilde{r}$ priced by the algorithm has an inconclusive reduced cost. That is,

$$E^K(c_{\tilde{r}}) - \sum_{i \in C} \alpha_{\tilde{r}}^i \rho_i < 0 \leq \mathbb{E}(c_{\tilde{r}}) - \sum_{i \in C} \alpha_{\tilde{r}}^i \rho_i,$$

where ρ is an optimal dual solution of the restricted master solved with the current route set $\tilde{R}$. This implies for any route $r \in R_K$ that

$$\sum_{i \in C} \alpha_r^i \rho_i - \mathbb{E}(c_r) \leq \sum_{i \in C} \alpha_r^i \rho_i - E^K(c_r) \leq \sum_{i \in C} \alpha_{\tilde{r}}^i \rho_i - E^K(c_{\tilde{r}}) \leq \mathbb{E}(c_{\tilde{r}}) - E^K(c_{\tilde{r}}) = \sum_{\ell=K+1}^{n_{\tilde{r}}-1} H_{\tilde{r}}^{\ell} \leq \delta_K,$$

where the second inequality follows because $\tilde{r}$ is the route produced by the approximate pricing problem. □

Like UCA, this algorithm has an *a priori* gap guarantee.

Corollary 7. *Suppose $K_0 = K$ and let y^* be optimal for (2.5) with respect to R_K. The solution produced by FCA has objective value no greater than*

$$\sum_{r \in R_K} y_r^* (\mathbb{E}(c_r) + \delta_K) = f(R_K) + \delta_K \sum_{r \in R_K} y_r^*.$$

As with UCA, a similar guarantee applies when $K_0 < K$, except the set of routes we optimize over may be larger than R_K, as it contains any routes with smaller cycles that the algorithm included in earlier iterations.

Corollary 8. *Using FCA within a B&P algorithm yields an integer solution with expected cost within an additive gap $\delta_K n$ of optimal. The factor of n can be substituted by any known tighter bound on the number of routes used by an optimal solution.*

Proof. The gap given in Corollary 7 depends on an optimal solution of the LP; in B&P, this solution would vary by node, so we can only claim an overall gap that is guaranteed no smaller than the gap at any node. Since we may assume $\sum y_r \leq n$ without loss of optimality, the result follows. If we have an upper bound on $\sum y_r$ that is tighter than n, the same argument applies with this bound. □

Corollary 9. *To achieve any desired additive optimality gap $\epsilon \geq 0$ for the VRP-PC via FCA within B&P, it suffices to take $K = O(\log(n/\epsilon))$.*

Proof. The proof is identical to Corollary 3, except we start with $\delta_K n \leq \epsilon$. □

2.4 Computational Study

In this section, we test both of our algorithms implementing a proof of concept for the VRP-PC, including a particular hard sequence-dependent constraint. We also study the empirical convergence of the expected cost approximation for UCA and FCA, the algorithms' running times, and how their performance is affected by model parameters.

2.4.1 Instances

We test our algorithms on VRP-PC instances based on the Solomon instances of the VRP with time windows (VRPTW) [38]. When customers may or may not be present, it might not be practical to establish hard time window constraints, as in the VRPTW. Nonetheless, we carry out our experiments with these instances for two reasons: First, time windows are a type of route- or sequence-dependent constraint that is difficult to capture with arc-based formulations, where the corresponding relaxations are known to be weak. Second, the Solomon instance family is a generally accepted benchmark widely used by researchers to test VRP algorithms.

The Solomon instances have 100 customers each and are divided into three categories: C (clustered), R (uniformly distributed) and RC (mix of R and C). Using 5 instances each from type C (C101 to C105) and from R (R101 to R105), we create VRP-PC instances with 15, 25 and 40 customers. Instances with a given number of customers have a different sequence of customers with respect to the original Solomon instance. For example, we create two 40-customer instances from each original Solomon instance, one considering customers 1 to 40 and another with customers 41 to 80. Because our largest instances have 40 customers, we reduce the vehicle capacity from 200 to 80. All instances have the original depot location.

With this procedure, we respectively obtain 60, 40 and 20 "base" deterministic

VRPTW instances with 15, 25 and 40 customers. We then tested our B&P implementation on each of these deterministic instances, eliminating two 25-customer and four 40-customer instances we could not solve to optimality within a 6-hour time limit; this reduction allows us to compare our VRP-PC results against the corresponding deterministic solution, and also lets us focus more on the difficulty increase brought on by the problem's probabilistic nature, rather than on the challenges it inherits from the VRPTW. After this elimination, each remaining base deterministic instance generates three VRP-PC instances, each with a different uniform customer probability $p \in \{0.5, 0.7, 0.9\}$, yielding a total of 342 instances.

2.4.2 Experiments

We tested both algorithms, UCA and FCA, on each instance with $K_0 = 1$ and $K \in \{1, \ldots, 5\}$, applying a 6-hour time limit. In total, this involves $1,710$ runs of each algorithm on the different instances. We ran the experiments in the Georgia Tech ISyE computing cluster, which uses HTCondor to manage its jobs, on an Intel Xeon E5-2603 (1.80GHz) machine with up to 10Gb of RAM, and using CPLEX 12.4 as LP solver. As a reference, when we run our B&P implementation on the original deterministic instances with 100 customers, we can solve several instances to optimality (of both type C and R) in a few minutes. This gives further indication that the main computational challenge we face stems from the optimization of expected route cost, and also suggests, as expected, that optimizing the VRP-PC is significantly more difficult than its deterministic counterpart.

In our first set of results, shown in Table 2.2, we report the average maximum value of K our B&P algorithms were able to solve to optimality and within a 5% and 10% relative gap. (The gap measured here is between the best integer solution and best bound found by the B&P tree.) For example, for instances with 15 customers and probability 0.5, the average largest K value for which our UCA B&P algorithm

can report a 0% relative gap within the time limit is 4.45. The corresponding average for FCA is 4.52.

Table 2.2: Average largest K solved by each algorithm within a certain relative gap.

n	p	UCA			FCA			Total Runs
		0%	5%	10%	0%	5%	10%	
15	0.5	4.45	4.83	4.95	4.52	4.88	4.95	900
	0.7	4.67	4.9	4.97	4.78	4.83	4.98	
	0.9	4.87	4.9	5	4.87	4.93	5	
25	0.5	3.37	4.08	4.68	3.55	4.29	4.71	570
	0.7	3.45	4.11	4.74	3.58	4.16	4.76	
	0.9	3.71	4.11	4.58	3.68	4.37	4.71	
40	0.5	2.5	3.19	4.38	3	3.94	4.19	240
	0.7	2.38	3.69	4.25	2.62	3.88	4.25	
	0.9	2.62	4.38	4.5	2.56	4.19	4.44	

From the table we see the clear impact the number of customers n has on the parameter K we can use when running the algorithms to optimality. For 15 customers, K can be 4 or 5; for 25 customers, K can be about 3 or 4; and for 40 customers K can be 3 and sometimes 2. Similarly, the customer realization probability p affects this average K; as we might expect, the larger the probability, the larger K can be, though there are some exceptions. The choice of K has two separate consequences. First, it helps determine the *a priori* and *a posteriori* additive gap guarantees of solution quality we get from UCA and FCA, since they depend on the number of terms we consider in the expected value approximation E^K. Second, it impacts the problem's difficulty through the pricing problem, which is solved as an SPPRC-K.

In Figure 2.1 we depict the average UCA *a priori* and *a posteriori* gap guarantees as a function of K for the three different customer probabilities $p \in \{0.5, 0.7, 0.9\}$. For each value of K and p, we include in the average only those instances we were able to solve to optimality in the B&P algorithm. We report these gaps as relative distances to the optimal solution; the gaps decay quite significantly for higher K values, so we present them in natural logarithmic scale (*i.e.*, as powers of e). For example, for $p = 0.5$ and $K = 3$ the average *a priori* gap is roughly $1/e \approx 37\%$,

meaning we can guarantee before running the algorithm that the expected cost of the solution returned by UCA is at worst roughly a third costlier than the optimum, on average.

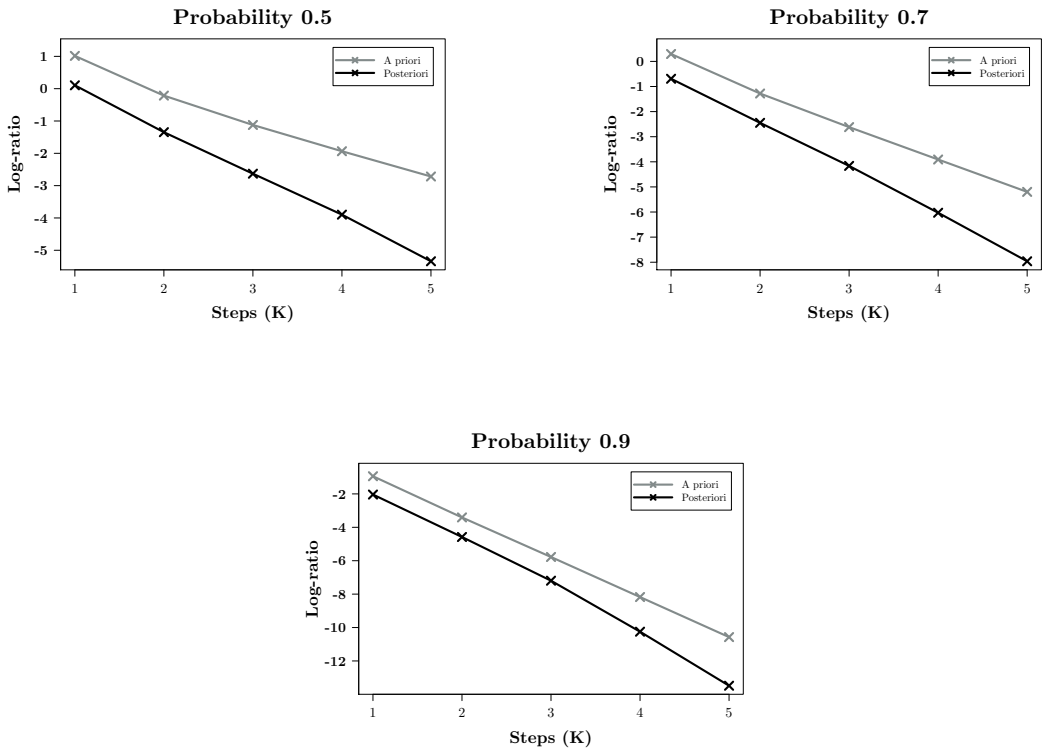

Figure 2.1: Average *a priori* and *a posteriori* guarantees by probability.

These results indicate how quickly both guarantees converge to zero as we increase K. For any probability, $K = 4$ or $K = 5$ more than suffice for either algorithm to return a solution with expected cost very close to optimal. Furthermore, for $K \geq 3$ the difference between *a priori* and *a posteriori* gaps is already within 10%-20% or less, with the *a priori* guarantee at worst being about 30% from optimal.

Figure 2.2 shows the convergence between the UCA and FCA solutions' expected cost and UCA's bound in an absolute scale, as a function of the algorithm parameter K, plotted by customer realization probability. The averages here include only those instances for which we were able to run the B&P algorithm to optimality for all values of K, in order to make the comparison using a fixed set of instances. The figure plots

the average of the the UCA optimal value measured with approximation E^K, which is a lower bound on the optimal expected cost; it also plots the UCA and FCA solutions' exact expected cost, UCA + Post and FCA. We observe that the bound provided by UCA converges very fast to the optimum, especially when the customer realization probability is high. Moreover, the expected cost of either algorithm's solution is quite close to the optimum, even for $K = 1$.

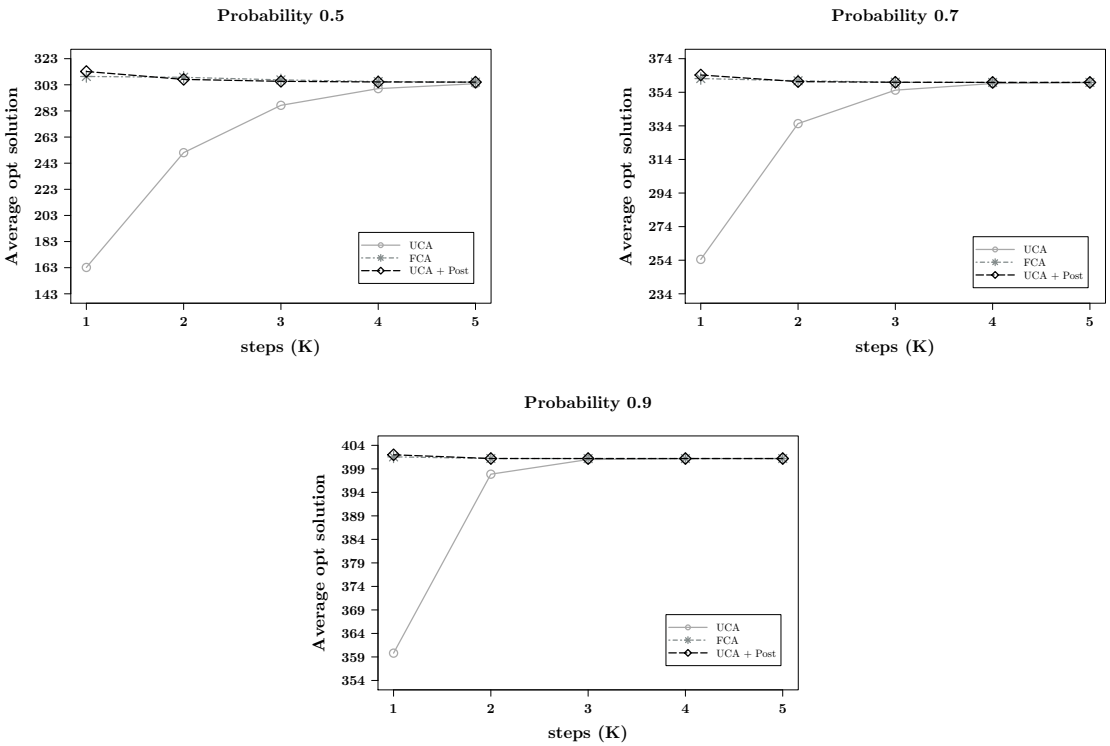

Figure 2.2: Convergence of UCA lower bound and solutions' expected cost, by probability.

Table 2.3 presents the geometric mean of relative gaps across instances of three solutions: the UCA solution (with exact expected cost), the FCA solution, and the expected cost of the deterministic VRPTW solution that ignores probabilities and minimizes the cost as if all customers will require visits. The gaps are calculated with respect to the best possible lower bound computed by UCA for any value of K, where we include all instances we could solve for that K; this means the number of instances included in an average may vary by value of K. We show results for $p = 0.7$, with

similar tables for $p \in \{0.5, 0.9\}$ in the Appendix, and separate results by instance type (C and R); recall that instances of type C have clustered customer locations, while type-R instances have uniformly distributed customer locations.

Table 2.3: Average relative gap of solution expected cost for UCA, FCA and deterministic problem, for realization probability 0.7.

Customers	Steps (K)	Type C			Type R		
		UCA Exact / Best LB	FCA / Best LB	Deter / Best LB	UCA Exact / Best LB	FCA / Best LB	Deter / Best LB
15	0	1.55%	1.45%	1.59 %	0.44%	0.53%	0.88 %
	1	0.49%	0.53%		0.39%	0.55%	
	2	0.25%	0.27%		0.41%	0.42%	
	3	0.25%	0.25%		0.41%	0.41%	
	4	0.25%	0.25%		0.41%	0.41%	
25	0	2.32%	2.55%	3.29 %	0.86%	1.09%	1.62 %
	1	2.83%	3.01%		1.15%	1.39%	
	2	2.97%	2.89%		1.09%	1.11%	
	3	2.57%	2.52%		0.97%	0.97%	
	4	2.04%	2.04%		1.01%	1.06%	
40	0	3.93%	3.93%	4.58 %	1.41%	1.41%	1.41 %
	1	4.58%	4.58%		1.11%	1.28%	
	2	4.58%	4.58%		1.06%	1.13%	
	3	4.58%	4.58%		0.07%	0.07%	
	4	3.93%	3.93%		0.07%	0.07%	

In all cases, the expected cost of solutions produced by UCA and FCA are within 5% of optimal on average, often much closer, and both are consistently better than the solution given by the deterministic instance. Either algorithm can produce the better solution on a particular instance; we detect no clear pattern of one producing better solutions than the other, but overall UCA has a slight advantage. Unsurprisingly, the number of customers n impacts the solutions' gap, with bigger instances having larger gaps. More interestingly, the instance type significantly affects the solutions' quality, with gaps for type R on average much tighter than for type C. Unlike the results summarized in Figure 2.2, here we do not always see a monotonically decreasing gap as K increases, but this is a result of including different sets of instances for different values of K; in general, for a given instance we tend to see better solutions with larger

K.

To further explore the benefit of optimizing with respect to expected costs, Figure 2.3 plots the percentage of instances where UCA and FCA obtain solutions with lower expected cost than the solution of the deterministic VRPTW. For example, when the customer realization probability is 0.5 and $K \geq 4$, both UCA and FCA produce better solutions in about 60% of the tested instances. The plots also emphasize that the realization probability affects this improvement percentage; for larger probabilities, it is harder for the algorithms to improve on the deterministic solution, presumably because this solution is already close to optimal. In particular, when the probability is 0.9, we only find a better solution roughly one quarter of the time, and this improvement occurs already with $K \geq 2$. In general, we also observe that UCA is slightly better than FCA at improving over the deterministic solution.

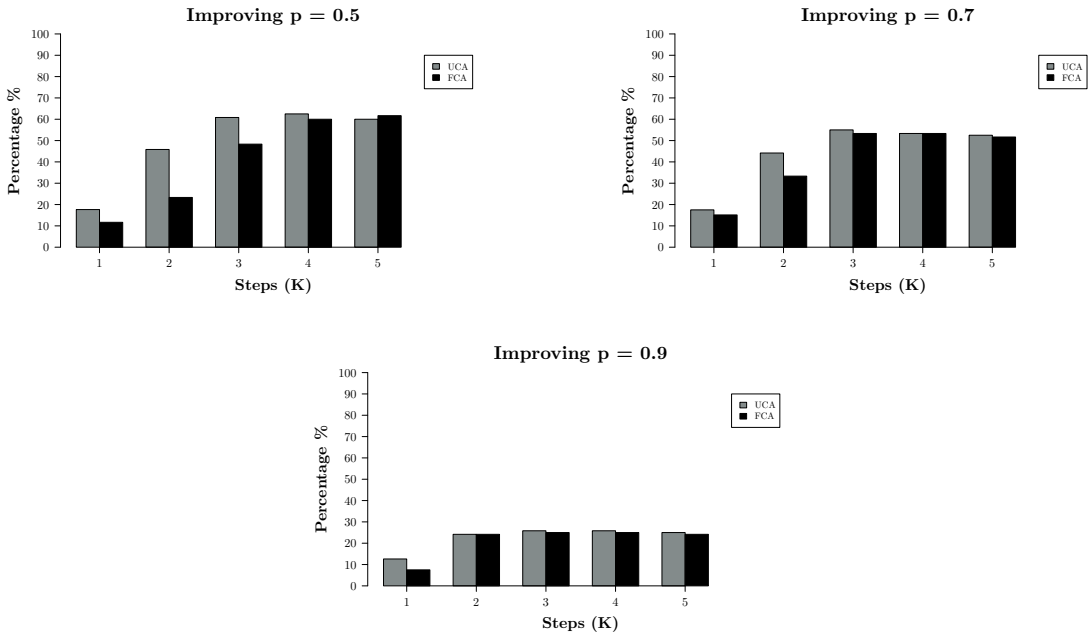

Figure 2.3: Percentage of instances where UCA and FCA improve on the deterministic solution.

Finally, even when the difference in expected cost between the deterministic solution and our algorithms' solutions is not large, the structure of the corresponding

solutions changes drastically, and may have operational implications, as discussed next in Section 2.4.3.

In Table 2.4 we present the average number of B&B nodes and the average number of routes generated per node for the UCA and FCA algorithms over instances with 15 customers, since those instances are solved closer to optimality. Both algorithms generate a similar magnitude of B&B nodes, but the FCA algorithm shows a more clear increasing trend as p decreases. The average number of routes generated by node is slightly smaller for FCA and for R type instances. Also, for C type instances, this number decreases as the customer realization probability increases.

Table 2.4: Average number of B&B nodes and routes generated by UCA and FCA algorithms in instances with 15 customers.

Probability	UCA				FCA			
	Type C		Type R		Type C		Type R	
	Nodes B&P	Avg Routes	Nodes B&P	Avg Routes	Nodes B&P	Avg Routes	Nodes B&P	Avg Routes
0.5	2793.87	71.83	2843.65	42.86	3000.13	53.23	4797.33	38.38
0.7	3580.88	64.19	2356.56	46.79	2434.08	48.74	2678.5	40.09
0.9	2106.52	58.23	2323.48	49.95	1020.77	45.49	1265.13	40.82

2.4.3 Empirical Insights

In Figure 2.4, we plot the UCA solution and a deterministic solution assuming all customers will realize for an instance of type R with $n = 15$ and $p = 0.5$. The expected cost of the solution to the deterministic problem is 314.19, and it is plotted on the left. The expected cost of the UCA solution with $K = 5$ is 304.27, and it is plotted on the right. The difference in expected cost is only about 3%, yet the solutions are very different. Specifically, both solutions use 5 planned routes, but without repeating any, and the UCA solution's planned routes appear inefficient when viewed as deterministic routes; in particular, they include several crossings that the deterministic solution avoids. The results for this instance highlight the potentially counter-intuitive nature of probabilistic routing, where we may plan routes that appear inefficient, but whose expected cost is in fact lower than routes that appear cheaper.

We next plot the deterministic and UCA ($K = 3$) solutions for a type-C instance

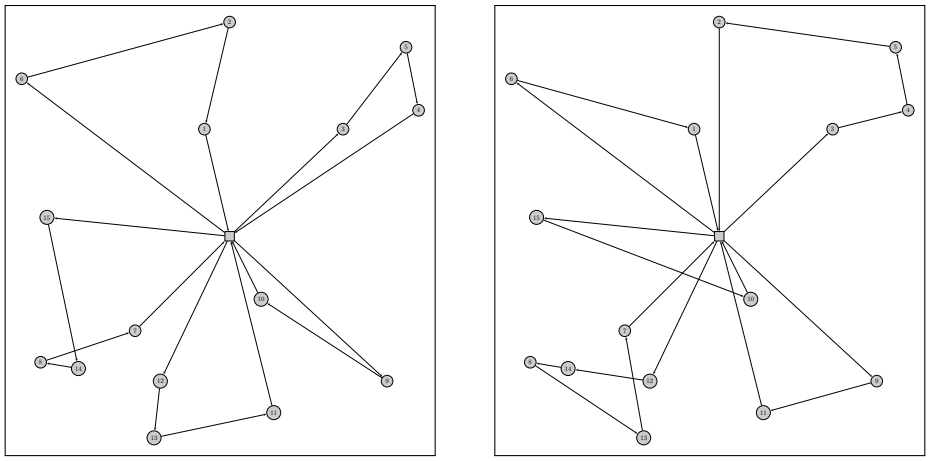

Figure 2.4: Deterministic (left) and UCA (right, $K = 5$) solutions for sample instance (type R) with $n = 15$ and $p = 0.5$.

with 40 customers (Figure 2.5). We observe that some routes are identical in both solutions, while others differ. As in the previous example, routes do not frequently cross for the deterministic solution, while they often do for UCA. Intuitively, the marginal deterministic cost increase generated by routes that cross more frequently is significantly less than the expected cost savings generated by probabilistic customers when they are skipped. Take for example route "c": Its deterministic solution covers two clusters of two and three customers, respectively. The stochastic solution instead covers a cluster of four customers and an isolated customer; if this customer does not show up, the cost of this route decreases significantly, and it is much more likely that one customer is not present, as opposed to the likelihood of two not being present simultaneously. A similar argument can be made for route "f".

2.5 Conclusions

We have studied the VRP-PC, a broad class of routing problems with probabilistic customers, and proposed a new column generation and B&P framework, including two different algorithms, UCA and FCA. Both circumvent the difficulty of exactly pricing routes by using an approximate expected cost that under-estimates the true

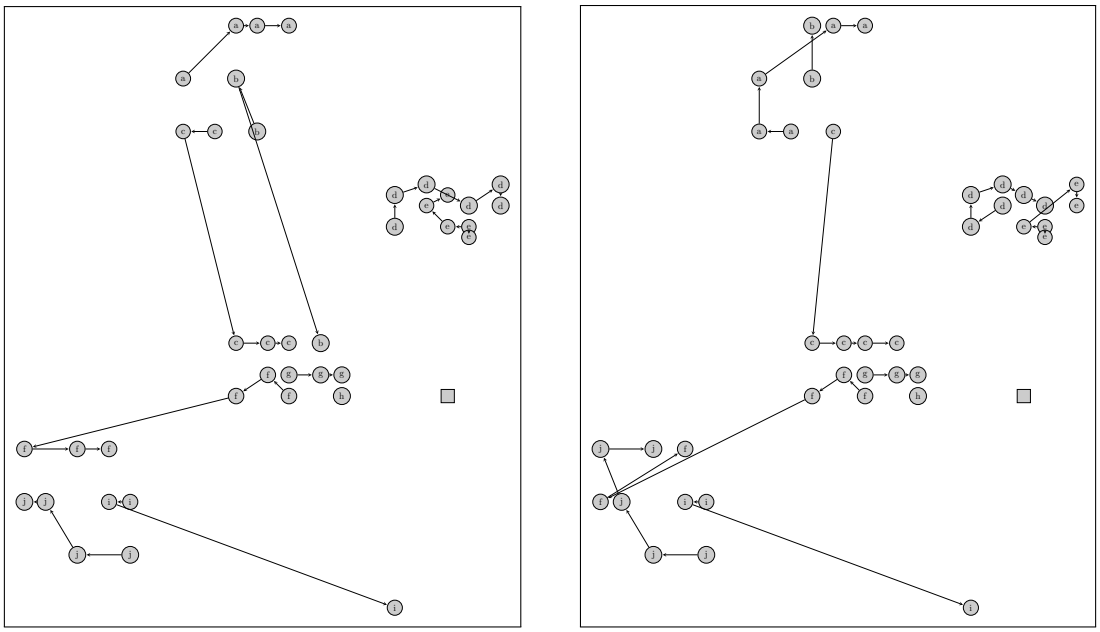

Figure 2.5: Deterministic (left) and UCA (right, $K = 3$) solutions for sample instance (type C) with $n = 40$ and $p = 0.5$.

expected cost. UCA optimizes with respect to the approximate expected cost, and thus provides an *a posteriori* lower bound to the optimal expected cost, in addition to giving a solution. FCA uses exact expected costs but may halt when some routes still have small negative reduced cost. Both algorithms have approximate optimality guarantees in the form of *a priori* additive gaps that depend on the precision of the expected cost approximation.

Our computational results suggest the *a posteriori* gap provided by UCA is much tighter than the theoretical *a priori* gap indicates. Furthermore, both gaps decrease quite rapidly as we increase the precision of the expected cost approximation in UCA or FCA; in our instances, an approximation with five steps ($K = 5$) or fewer suffices to get a negligible gap. However, the problem's difficulty is determined also by the number of customers and their realization probability; in particular, when the probabilities are large we can close the gap quickly. More generally, our results also suggest that UCA and FCA produce very good solutions, no more than 5% from optimal and usually much better. Both algorithms improve upon the solution of the

deterministic instance in many cases, although the improvement is not always large in terms of relative gap. However, even in these cases the structure of the resulting solution may change significantly.

Our results motivate several questions for future research. One option is to incorporate cutting planes into our B&P framework with UCA or FCA, which may allow us to increase the size and/or difficulty of instances we can optimize. Another possibility in this vein would be to use different relaxations of the ESPPRC, such as the ng-path relaxations introduced in recent years [60]; however, it is not immediately clear how our analysis or gap guarantees would extend here. Another improvement to our B&P method would be to include a dual stabilization technique; approaches like the *stabilization primal-dual* and *trust region* methods [61] could speed up computation times. An interesting question relates to combining our approach with approaches that optimize other related stochastic routing models. In particular, the use of chance constraints for resource consumption [54] is an interesting area with much potential for future work. Another idea would be to include a recourse action when a key resource is depleted, as in [16]; in this article each customer demand quantity is random and the vehicle is forced to pay a replenishment trip to the depot when it runs out of capacity. The challenge of such an approach would be to include such a recourse action cost within a pricing subproblem. More generally, the broad topic of column generation in probabilistic and *a priori* optimization offers many challenging questions for the research community.

CHAPTER 3
THE CONTINUOUS TIME INVENTORY ROUTING PROBLEM

3.1 Introduction

The Inventory Routing Problem (IRP) integrates inventory management, vehicle routing, and delivery scheduling decisions. The IRP arises in the context of Vendor Managed Inventory (VMI), in which a supplier makes the replenishment decisions for products delivered to its customers. The variant of interest in this chapter was introduced in the seminal chapter by [2]. Critical characteristics of this variant are that only transportation costs are considered and that the system evolves in continuous time. That is, the amount of product that can be delivered at a customer at a particular point in time depends on the storage capacity and inventory at that point in time (which depends on the initial inventory, the product usage rate and the time elapsed since the start of the planning period, and the amount of product delivered since the start of the planning period). As a consequence, delivery times have to be scheduled carefully and vehicle travel times have to be accounted for accurately. This contrasts with the majority of the variants of the IRP considered in the literature, where the planning horizon is partitioned into periods and it is assumed that delivery routes take place at the start of the period, product consumption takes places at the end of the period, and that both happen instantaneously. For more comprehensive introductions to and discussions of the IRP, see [62] and [5].

The variant considered in this chapter (and by [2]) is motivated by the IRP encountered by companies in the liquid gas industry, e.g., Air Liquide (`www.airliquide.com`), Air Products (`www.airproducts.com`), and Praxair (`www.praxair.com`). These companies produce liquefied gases, e.g., liquid oxygen, liquid nitrogen, or liquid ar-

gon, and install tanks at their customers' premises and guarantee minimum product availability at any time, Customers use (consume) product at a certain rate (which can differ at different times of the day) often 24 hours per day (e.g., liquid oxygen in hospitals.) Thus, the amount of product that can be delivered to the tank changes at the same rate. The companies continuously monitor product usage and tank inventory levels so that they can produce cost-effective delivery schedules that meet their service commitments (i.e., the guaranteed minimum product availabilities). In practice, the companies tend to have customers that require multiple deliveries per day as well as customers that require only one or two deliveries per week. As a consequence, the use of a continuous time variant of the IRP is most appropriate in these settings, i.e., provides the most accurate representation of the system. Also relevant is the fact that the company contracts typically specify that the customers own/purchase the product upon delivery, which means that the companies do not have to consider product holding costs at the customer. This variant of the IRP has attracted attention in the past, e.g., [4], [63], and [64, 65], and, more recently, was considered interesting and challenging enough to form the ROADEF/EURO 2016 Challenge (for more information, see `www.roadef.org/challenge/2016/en/sujet.php`) with real-life data provided by Air Liquide.

Solving instances of any variant of the IRP is challenging [66]. Integer programming techniques have been used for the "period" variants, e.g., branch-and-cut [67, 68] and branch-and-cut-and-price [67, 69]. The most advanced and successful of these can now solve instances with up to 50 customers and up to 5 vehicles. For the "continuous time" variant, no optimization algorithms exist, to the best of our knowledge, but lower bounding techniques have been developed in [70].

As we shall find in this study, modeling inventory in continuous time and accurately modeling vehicle travel times makes for a particularly challenging IRP. Indeed, optimization problems over continuous time, in general, have been found to be difficult

to solve to optimality. Compact models that use continuous variables to model time have weak linear programming (LP) relaxations. Their solution with current integer programming solver technology is limited to only small instances. Extended formulations, with binary variables indexed by time, have much stronger relaxations, but (tend to) have a huge number of variables. Such formulations rely on a *discretization* of time, which introduces approximation. Recently, [8] introduced a dynamic discretization discovery algorithm for solving the continuous time service network design problem, which uses extended integer programming formulations. The key to the approach is that it discovers exactly which times are needed to obtain an optimal, continuous-time solution, in an efficient way, by solving a sequence of (small) integer programs. The integer programs are constructed as a function of a subset of times, with variables indexed by times in the subset. They are carefully designed to be tractable in practice, and to yield a lower bound (it is a cost minimization problem) on the optimal continuous-time value. Once the *right* (very small) subset of times is discovered, the resulting integer programming model yields the continuous-time optimal value.

In this chapter, we explore and demonstrate the potential of dynamic discretization discovery algorithms for the continuous time variant of the IRP, *CIRP*. The aim of our research is twofold. First, we want to develop an optimization algorithm for this important variant of the IRP, and by doing so hope to stimulate others to take up this challenge as well. Second, we want to demonstrate that dynamic discretization discovery algorithms can be developed and useful for problems other than service network design, and by doing so, again, hope to stimulate others to start using and advancing this approach.

Our contributions in this chapter are both theoretical and algorithmic. We

- investigate the problem of minimizing the number of vehicles needed to obtain a feasible delivery plan, showing that it is strongly NP-hard, but that in the

case of a single customer, it can be solved in pseudo-polynomial time,

- develop a mixed integer programming model for the CIRP over a given, uniform, discretization of time,
- prove that if the data for an instance is rational, then it has an optimal solution in which all delivery times at customers are rational, which yields, as a consequence, that the mixed integer programming model can, in theory, provide optimal solutions to the CIRP, by taking the discretization corresponding to these rationals,
- adapt the mixed integer program to yield lower bounds on the optimal CIRP value, which can be achieved in practice by using a sufficiently coarse discretization,
- propose model enhancements that strengthen both the resulting lower bound and the mixed integer programming formulation itself,
- develop two alternative approaches to finding feasible solutions, one using an adaptation of the mixed integer program and the other based on repairing solutions to the lower bound model, and
- carry out a computational study to assess the performance of these ideas, in practice.

Our study shows that with only a partial implementation of a dynamic discretization discovery algorithm, we have been able to optimally solve instances with up to 15 customers and have been able to obtain provably high-quality solutions for many others. Even though these results are notable, they also point to areas for further research and improvement.

The remainder of the chapter is organized as follows. In Section 3.2, we formally introduce the continuous time inventory routing problem and some of the character-

istics that distinguish it from period-based inventory routing problems. In Section 3.3, we highlight the challenges associated with determining the minimum number of vehicles required to produced a feasible delivery plan. In Section 3.4, we provide a mixed integer programming formulation for the continuous time inventory routing problem over a given discretization. In Section 3.5, we discuss how the mixed integer programming formulation can be modified to produce a lower bound on the cost of an optimal delivery plan. In Section 3.6, we outline two approaches for constructing feasible delivery plans. In Section 3.7, we present and discuss the results of an extensive computational study. Finally, in Section 3.8, we describe the next steps towards a full-scale dynamic discretization discovery algorithm for the continuous time inventory routing problem.

3.2 The Continuous Time Inventory Routing Problem

We consider a vendor managed resupply environment in which a company manages the inventory of its customers, resupplying a single product from a single facility.

Each customer i in the set $N = \{1, \ldots, n\}$ of customers has local storage capacity $C_i > 0$, uses product at a constant rate $\hat{u}_i > 0$, and has initial inventory $I_i^0 > 0$ at the start of the planning period. Note that a customer uses product at a given rate, i.e., a customer consumes a certain amount of product per unit of time. The planning horizon is $H > 0$. The company employs a fleet of m homogeneous vehicles, each with capacity $Q > 0$, to deliver product to its customers. A delivery route has to start and end at the company's facility and has to be completed before the end of the planning horizon. Unlike many inventory routing problem settings, here vehicle routes are not restricted to start and end in a single time period. Indeed, the setting we study here is based on continuous time, not subdivided into periods. Travel times $\hat{\tau}_{ij} > 0$ and travel costs $c_{ij} \geq 0$ between every pair of locations i and j, $i \neq j$, for $i, j \in N_0 = \{0, 1, \ldots, n\}$, where 0 denotes the location of the company's facility,

are known. Travel times are assumed to satisfy the triangle inequality. There is no inventory holding cost.

We allow a vehicle to wait at a customer location and to make multiple deliveries while at the customer's premises. This may be beneficial as it allows delivery of more than the customer's storage capacity without the need for an intervening trip to the depot. (We provide an illustration of this point later.) In practice, a customer may have sufficient space for several vehicles to wait at their premises, but usually at most one vehicle can deliver product at a time. Although the time needed for a vehicle to deliver product may depend on the quantity to be delivered, there is usually a substantial overhead time needed to engage and disengage the delivery equipment. Thus the delivery time can be reasonably well approximated by a constant (possibly customer-dependent) length of time. This situation can be modeled by the use of two locations for each customer, one for parking and one for making a delivery at the customer, with the latter constrained to allow at most one vehicle at a time. However, given the complexity of the ideas we wish to discuss in this chapter, we make the simplifying assumption that all locations have the single-vehicle constraint: we assume that it is *not* possible for multiple vehicles to visit the same customer at the same time. We further assume that product delivery at a customer site is instantaneous. It is not difficult to extend the ideas we present here to account for multiple vehicles waiting at a customer and constant delivery time. We also assume that there is no cost for waiting. (In the liquid gas industry, drivers are salaried employees, so their time, for the purpose of this model, can be considered a sunk cost.) Finally, we can assume, without loss of generality, that customers are served by a finite set of deliveries, occurring at a finite number of time points during the planning period.

Vehicles are assumed to be at the company's facility at the start of the planning period and have to be back at the company's facility at the end of the planning period.

However, vehicles can make multiple trips during the planning period. We assume that the loading of a vehicle at the company facility is instantaneous.

We assume that the company does not incur any holding cost for product, either at the company facility or at any of the customer sites. We also assume that the company facility always has sufficient product to supply customers; it does not have either production or storage capacity constraints.

The goal is to find a minimum cost delivery plan that ensures that none of the customers runs out of product during the planning period. A delivery plan specifies a set of vehicle itineraries, each of which consists of a sequence of trips/routes that start and end at the company facility within the time horizon, to be performed by a single vehicle. Each route specifies a departure time from the facility, a sequence of customer deliveries, and, for each, the delivery time and quantity delivered to the customer at that time. We refer to this problem as the Continuous Time Inventory Routing Problem (CIRP).

We start by presenting some observations about optimal solutions to instances of the CIRP.

- There exist instances of the CIRP in which it is beneficial to visit a customer more than once on a delivery route. Consider, for example, an instance with two customers, storage capacities $C_i = 2$ for $i = 1, 2$, usage rates $\hat{u}_i = 1$ for $i = 1, 2$, initial inventories $I_1^0 = 1$ and $I_2^0 = 2$, a single vehicle with capacity $Q = 5$, travel times and costs $\hat{\tau}_{01} = c_{01} = \hat{\tau}_{12} = c_{12} = 1$ and $\hat{\tau}_{02} = c_{02} = 2$, and a time horizon $H = 4$. The optimal solution has a single route, of cost 4, visiting Customer 1 at time 1, delivering 2 units of product, visiting Customer 2 at time 2, delivering 2 units of product, and visiting Customer 1 again at time 3, delivering 1 unit of product; see Figure 3.1. This must be optimal, since any feasible solution must visit Customer 2 at least once, at a cost of 4.

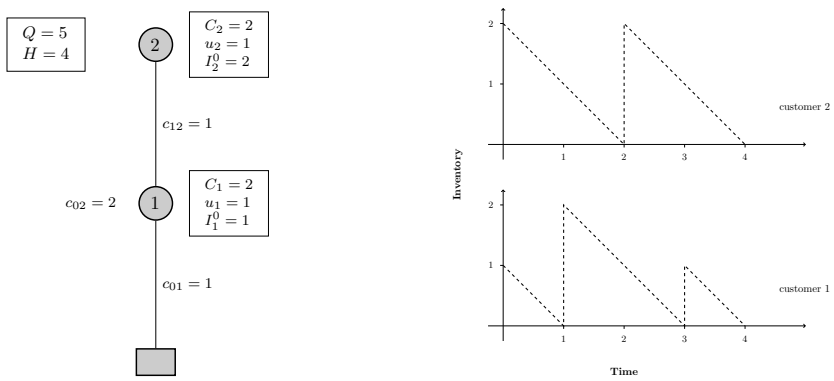

Figure 3.1: Instance for which it is optimal to visit a customer more than once on a route.

- A similar example, with only the single customer, Customer 1, illustrates that an optimal solution may require the vehicle to wait at the customer to make multiple deliveries. With the same parameter values as above, but without Customer 2, any optimal solution is a single route, visiting Customer 1 at time 1 to deliver some product, for example, 2 units, and then waiting, for example, until time 2, to deliver the 1 unit remaining of the total 3 units required; see Figure 3.2.

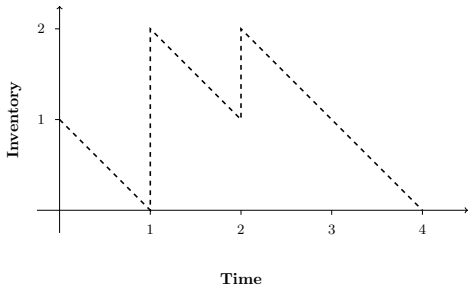

Figure 3.2: Instance for which it is optimal to waiting at a customer and make multiple deliveries.

- There exist instances of the CIRP in which all parameters are integer valued (i.e., capacities, usage rates, initial inventories, and travel times), but for which there exists no optimal solution that delivers product at customers only at integer times. Consider, for example, an instance with two customers, having

storage capacities $C_1 = 7$, $C_2 = 5$, usage rates $\hat{u}_i = 2$, $i = 1, 2$, and initial inventories $I_1^0 = 3$, $I_2^0 = 5$. There is a single vehicle with capacity $Q = 12$. The travel times and costs are identical and symmetric: $\hat{\tau}_{01} = c_{01} = \hat{\tau}_{12} = c_{12} = \hat{\tau}_{02} = c_{02} = 1$. The time horizon is $H = 5$. The optimal solution has a single route of cost 3, visiting Customer 1 at time 1.5, delivering 7 units of product, and visiting Customer 2 at time 2.5, delivering 5 units; see Figure 3.3. This solution is optimal because any feasible solution must visit each customer at least once, and the cheapest way to do this is with a single route. There cannot be an optimal solution in which the deliveries take place only at integer times, since Customer 1 must have a delivery on or before time 1.5, when it runs out of product, and at time 1, Customer 1 only has capacity for 6 units of product. If the remaining 1 unit it requires is to be delivered without incurring extra cost, the vehicle must wait at Customer 1 until time 2 to deliver this unit, at which time it is too late to reach Customer 2 by time 2.5, when Customer 2 runs out of product. Thus the solution with delivery to Customer 1 and time 1.5 and Customer 2 and time 2.5 is the unique optimal solution.

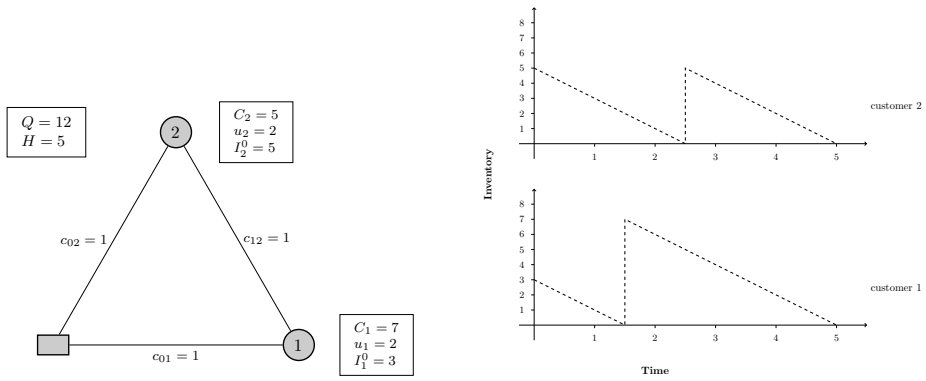

Figure 3.3: Instance with integer data for which the optimal solution has non-integer delivery times.

The last example illustrates the challenge in using discretization to approach the CIRP. An approximation in which vehicle departure and delivery times are restricted

to integers would yield an optimal solution costing twice that of the true, continuous time optimal solution. Even an approximation with a discretization into periods of length 0.2, needing 15 time periods, still yields a factor of two error in the optimal value. To obtain the optimal continuous time solution in this case one may either use 6 periods of length 0.5, or 30 periods of length 0.1. This observation highlights another challenge: clearly the quality of the approximation from discretization is not monotonically improving in the granularity of the discretization.

3.3 Minimizing the number of vehicles

Considering the minimum number of vehicles required to produce a feasible delivery plan also reveals the complexity of the CIRP, as shown in the next two propositions.

Proposition 10. *The problem of finding the minimum number of vehicles required to produce a feasible delivery plan for a CIRP instance is strongly NP-hard.*

Proof. We show this with a reduction from the bin packing problem (BPP). Consider a BPP instance with a set of items $\{1, \ldots, n\}$, each with size a_i for $i = 1, \ldots, n$, and a set of bins $\{1, \ldots, m\}$, each with capacity V. Without loss of generality, we may assume $2 \leq a_i \leq V$ and integer for $i = 1, \ldots, n$.

Let the corresponding CIRP instance have time horizon $H = V$ and a fleet of homogeneous vehicles with capacity $Q = n$. For each item $i \in \{1, \ldots, n\}$ in the BPP instance, there is a corresponding customer $i \in N$ in the CIRP instance. The travel time $\hat{\tau}_{0i}$ from the depot to customer i is $\frac{1}{2}a_i$ for $i \in N$. The travel time $\hat{\tau}_{ij}$ from customer i to customer j is $\hat{\tau}_{ij} = \hat{\tau}_{i0} + \hat{\tau}_{0j}$ for $i, j \in N, i \neq j$. Furthermore, for each customer $i \in N$, let the storage capacity be $C_i = H$, the initial inventory be $I_i^0 = H - 1$, and the usage rate be $u_i = 1$. Note that this implies that all customers have to be visited at least once.

Let m^* be the minimum number of vehicles required to produce a feasible delivery plan for the CIRP instance. Let $S_k \subseteq N$ be the set of customers visited by the k^{th}

vehicle, $k = 1, ..., m^*$. For each $k = 1, \ldots, m^*$, let the items corresponding to the customers in S_k be assigned to Bin k. Because the total travel time for k^{th} vehicle is $\sum_{i \in S_k} 2\hat{\tau}_{0i} = \sum_{i \in S^k} a_i \leq H = V$, this assignment is feasible. Thus, there exists a solution to the bin packing problem with m^* bins. Conversely, let m^* be the number of bins in an optimal solution to the BPP instance and let S_k be the items assigned to Bin k for $k = 1, \ldots, m^*$. The corresponding CIRP solution in which the customers in S_k are visited by vehicle k is feasible (note that by the definition of the travel times, the order in which customers are visited is immaterial). □

It is not unexpected that finding the minimum number of vehicles required to produce a feasible delivery plan for a CIRP instance is NP-hard, but that finding the minimum number of vehicles required to produce a feasible delivery plan for a single-customer CIRP instance is also non-trivial may come as a surprise. Below, we provide a pseudo-polynomial time algorithm for finding the minimum number of vehicles required to produce a feasible delivery plan for a single-customer CIRP instance. It is still an open question whether a polynomial time algorithm exist, although we conjecture that it does.

In the remainder of this section, we use τ to denote the travel time from the depot to the (single) customer, u to denote the usage rate of the customer, and I to denote the initial inventory of the customer. If there is no limit on the number of vehicles available, then the problem has a feasible solution if and only if $C \geq I \geq \tau u$ and either $I \geq Hu$ or $H \geq 2\tau$.

Lemma 2. *If a single-customer CIRP instance has a feasible solution in which m vehicles make a total of n visits to the customer, then there is a feasible solution in which the vehicles make the trips in round-robin fashion, so Vehicle 1 makes trips $1, m+1, 2m+1, \ldots$, Vehicle 2 makes trips $2, m+2, 2m+2, \ldots$, etc. In general, Vehicle k makes the $(rm+k)$th trip for $r = 0, 1, 2, \ldots \lfloor (n-k)/m \rfloor$.*

Proof. Since there is only a single customer, all the trips are the same, depot-customer-depot, constituting a single visit to the customer, so we only have to assign the m vehicles to the n trips in the feasible solution. Furthermore, the vehicles are homogeneous, so any vehicle may be assigned to a trip without changing the quantity delivered on the trip, nor the time needed to complete it. (All vehicles require time τ to get to the customer and time τ to return from it.) Second, recall that no more than one vehicle can visit the customer at the same time, so no trip will arrive at the customer until after the preceding trip has departed, and trips are completely ordered in time.

Starting with the first trip in the solution, we may thus, without loss of generality, assign it to the first vehicle. Then we assign the second trip to the second vehicle, and so on. The first vehicle returns to the depot before the second vehicle returns (the second vehicle arrives at the customer only after the first vehicle finishes its delivery), the second vehicle returns to the depot before the third one, etc. Once we have assigned the first m trips, we consider a second trip for each vehicle, starting with the first vehicle. Since this vehicle was the first vehicle to return to the depot, it must be available for the $(m+1)$th trip. Then, if needed, we assign a third trip, and so on until we have assigned all trips to vehicles. (We refer to this process as "round-robin"). Thus, any feasible solution always has an ordered sequence of vehicles (from 1 to m) and each of them has a "similar" number of trips to perform. If m divides evenly into n, every vehicle performs exactly n/m trips; otherwise, $n \mod m$ vehicles perform $\lceil n/m \rceil$ trips and the remainder perform $\lfloor n/m \rfloor$ trips. □

Lemma 3. *Whether or not there is a feasible solution to a single-customer CIRP instance in which m vehicles make a total of n visits to the customer can be determined by solving a linear program.*

Proof. We can use the following linear program to determine whether a feasible solution with m vehicles making a total of n visits exists. By Lemma 2, we may assume

that the vehicles make the visits in round-robin fashion: we let $\bar{m} = n \bmod m$ and let $\ell_v = \lfloor \frac{n}{m} \rfloor + 1$ if $v \leq \bar{m}$ and $\ell_v = \lfloor \frac{n}{m} \rfloor$ if $v > \bar{m}$, so that ℓ_v denotes the last visit for vehicle v ($v = 1, \ldots, m$). Furthermore, let

- t^-_{vk} be the arrival time of the k^{th} visit of vehicle v,
- t^+_{vk} be the departure time of the k^{th} visit of vehicle v,
- q_{vk} be the quantity delivered during the k^{th} visit by vehicle v,
- I^-_{vk} be the inventory at the arrival time of the k^{th} visit of vehicle v,
- I^+_{vk} be the inventory at the departure time of the k^{th} visit of vehicle v, and
- ζ models the minimum time between visits by different vehicles at any customer.

Then the following linear program achieves our goal:

$$\max \zeta$$

$$t^-_{11} \geq \tau \tag{3.1a}$$

$$t^+_{\bar{m},\ell_{\bar{m}}} \leq H - \tau \tag{3.1b}$$

$$t^-_{vk} \geq t^+_{v-1,k} + \zeta \qquad v = 2, \ldots, m, k = 1, \ldots, \ell_v \tag{3.1c}$$

$$t^-_{1k} \geq t^+_{m,k-1} + \zeta \qquad k = 2, \ldots, \ell_1 \tag{3.1d}$$

$$t^-_{vk} \geq t^+_{v,k-1} + 2\tau \qquad v = 1, \ldots, m, k = 2, \ldots, \ell_v \tag{3.1e}$$

$$t^+_{vk} \geq t^-_{vk} + \frac{\max\{I^-_{vk} + q_{vk} - C, 0\}}{u} \qquad v = 1, \ldots, m, k = 1, \ldots, \ell_v \tag{3.1f}$$

$$I^+_{vk} = I^-_{vk} + q_{vk} - (t^+_{vk} - t^-_{vk})u \qquad v = 1, \ldots, m, k = 1, \ldots, \ell_v \tag{3.1g}$$

$$I^-_{vk} = I^+_{v-1,k} + (t^-_{vk} - t^+_{v-1,k})u \qquad v = 1, \ldots, m, k = 1, \ldots, \ell_v \tag{3.1h}$$

$$I^-_{1k} = I^+_{m,k-1} + (t^-_{1k} - t^+_{m,k-1})u \qquad k = 2, \ldots, \ell_1 \tag{3.1i}$$

$$\sum_{v,k} q_{vk} \geq Hu - I \tag{3.1j}$$

$$I^-_{vk} \geq 0 \qquad v = 1, \ldots, m, k = 1, \ldots, \ell_v. \tag{3.1k}$$

Constraint (3.1a) ensures that the vehicle making the first delivery can reach the customer before the start of the first delivery. Constraint (3.1b) ensures that the vehicle making the last delivery can return to the depot after completing the last delivery. Constraints (3.1c) and (3.1d) ensure that deliveries do not overlap, i.e., that the start of a delivery (which coincides with the arrival of a vehicle) does not happen until the preceding delivery has ended (which coincides with the departure of a vehicle). Constraints (3.1e) ensure the consecutive deliveries by the same vehicle properly account for the travel time to and from the depot. Constraints (3.1f) ensure that a vehicle does not depart from the customer until it was possible to transfer the entire delivery quantity into the customer's local storage. (These can easily be linearized.) Constraints (3.1g) are the inventory balance constraints associated with deliveries and account for any usage that may occur during a delivery (if the delivery is not instantaneous). Constraints (3.1h) and (3.1i) are the inventory balance constraints associated with the periods between consecutive deliveries. Constraints (3.1j) and (3.1k) ensure that the customer never runs out of product during the planning horizon. If the optimal objective value is positive, then there is a feasible solution, in which the requirement that no more than one vehicle is visiting a customer at a time is guaranteed by the objective; otherwise there can be no feasible solution. □

Proposition 11. *Determining the minimum number of vehicles required to produce a feasible solution to a single-customer CIRP instance can be done in pseudo-polynomial time.*

Proof. Observe that the number of deliveries required at the customer is at least $\lceil (Hu - I)/Q \rceil$, which implies that at most $\overline{m} = \lceil (Hu - I)/Q \rceil$ vehicles are needed (each vehicle making a single delivery). Observe too that a vehicle can make at most $\lfloor H/2\tau \rfloor$ deliveries, which implies that at least $\underline{m} = \lceil \lceil (Hu - I)/Q \rceil / \lfloor H/2\tau \rfloor \rceil$ vehicles are needed. These observations show that we can enumerate the possible combinations of number of vehicles m (i.e., $\underline{m} \leq m \leq \overline{m}$) and number of deliveries n (i.e., $m \leq n \leq m\lfloor H/2\tau \rfloor$

for a given number of vehicles m). For each combination, Lemma 3 shows that we can determine whether a feasible delivery schedule exists, by solving an LP with $\mathcal{O}(n)$ constraints and variables. This is pseudo-polynomial because the number of visits, n, depends polynomially on the planning horizon H (and thus is exponential in $\log(H)$). □

Conjecture 1. *Determining the minimum number of vehicles required to produce a feasible solution to a single-customer CIRP instance can be done in polynomial time.*

3.4 Optimal delivery plans

As mentioned in the introduction, we will use partially time-expanded network formulations to solve the CIRP. An implicit underlying assumption is that there exists a discretization of time such that an optimal solution to a time-expanded network formulation using this discretization results in a continuous time optimal solution. In Section 3.6, we prove that when the parameters of an instance are rational numbers, an optimal solution involving only rational numbers exists, which in turn implies that such a discretization exists. Therefore, consider a time discretization sufficiently fine that all time-based parameters (time horizon and travel times) are integer and so are all the times at which deliveries to and departures from a customer are made in an optimal solution. Specifically, suppose such a discretization is obtained by taking time intervals of length $\Delta > 0$ so that $H = T\Delta$ for some positive integer T and the discretization has T periods of length Δ. Under this discretization, the travel time from i to j is given by integer $\tau_{ij} = \hat{\tau}_{ij}/\Delta$ periods and the usage rate per period at customer i is given by $u_i = \hat{u}_i\Delta$. Under this discretization of time, we may safely restrict attention to solutions in which all deliveries to a customer occur and all vehicle departures occur at times in $\mathcal{T} = \{0, 1, \ldots, T-1\}$, where times are stated in units of periods of length Δ. We assume all travel times are non-negative, allowing travel time to be zero. In order to model waiting at a customer, and vehicles stationed at

the depot between trips, we introduce waiting time $\tau_{ii} = 1$ for $i \in N_0$. We take $c_{ii} = 0$ for all $i \in N_0$.

We now construct a mixed integer linear programming formulation with vehicles and product routed in a time-expanded network, with timed node set $\mathcal{N}^{\mathcal{T}} = \{(0, T)\} \cup (N_0 \times \mathcal{T})$ and timed arc set

$$\mathcal{A}^{\mathcal{T}} = \{((i, s), (j, t)) \in \mathcal{N}^{\mathcal{T}} \times \mathcal{N}^{\mathcal{T}} : (i, j) \in N_0 \times N_0, s + \tau_{ij} = t\}.$$

For a given instance the network $(\mathcal{N}^{\mathcal{T}}, \mathcal{A}^{\mathcal{T}})$ may be reduced by preprocessing to eliminate nodes and arcs that cannot appear in any feasible vehicle route. We use the notation $\delta^+(i, s)$ to represent the set of customers $j \in N_0$ with $((i, s), (j, s + \tau_{ij})) \in \mathcal{A}^{\mathcal{T}}$. Similarly, $\delta^-(j, t)$ is used to represent the set of customers $i \in N_0$ with $((i, t - \tau_{ij}), (j, t)) \in \mathcal{A}^{\mathcal{T}}$.

For each $((i, t), (j, t + \tau_{ij})) \in \mathcal{A}^{\mathcal{T}}$, let binary variable x_{ij}^t indicate whether a vehicle travels from location i to j departing from i at time t or not, and let variable w_{ij}^t indicate the amount of product that is transported from i to j at time t (zero when no vehicle travels from i to j departing from i at time t). Let variable y_i^t indicate the quantity of product delivered to customer $i \in N$ at time $t \in \mathcal{T}$ and z_i^t indicate the inventory level at customer i at time t. If a delivery takes place at time t, then z_i^t indicates the inventory value *after* the delivery. The model includes the possibility of customer deliveries at time 0, since we allow travel times that are zero. It is assumed that the initial inventory at each customer is enough to sustain it until a vehicle can arrive, i.e., $I_i^0 \geq \tau_{0i} u_i$ for all $i \in N$; otherwise the problem is infeasible.

We can now define the formulation:

$$\min \quad \sum_{(i,t)\in\mathcal{N}^{\mathcal{T}}} \sum_{j\in\delta^+(i,t)} c_{ij}x_{ij}^t$$

$$\text{s.t.} \quad \sum_{j\in\delta^+(i,t)} x_{ij}^t = \sum_{j\in\delta^-(i,t)} x_{ji}^{t-\tau_{ji}} \qquad (i,t)\in\mathcal{N}^{\mathcal{T}}\setminus\{(0,0),(0,T)\} \tag{3.2a}$$

$$\sum_{i\in\delta^-(0,T)} x_{i,0}^{T-\tau_{i,0}} = m \tag{3.2b}$$

$$\sum_{j\in\delta^+(i,t)} x_{ij}^t \leq 1 \qquad (i,t)\in\mathcal{N}^{\mathcal{T}} \tag{3.2c}$$

$$\sum_{j\in\delta^-(i,t)} w_{ji}^{t-\tau_{ji}} - \sum_{j\in\delta^+(i,t)} w_{ij}^t = y_i^t \quad (i,t)\in\mathcal{N}^{\mathcal{T}}, i\neq 0 \tag{3.2d}$$

$$0\leq w_{ij}^t \leq Qx_{ij}^t \qquad (i,t)\in\mathcal{N}^{\mathcal{T}}, j\in\delta^+(i,t) \tag{3.2e}$$

$$z_i^t = z_i^{t-1} + y_i^t - u_i \qquad i\in N, t\in\mathcal{T}\setminus\{0\} \tag{3.2f}$$

$$z_i^0 = I_i^0 + y_i^0 \qquad i\in N \tag{3.2g}$$

$$u_i \leq z_i^t \leq C_i \qquad i\in N, t\in\mathcal{T} \tag{3.2h}$$

$$0\leq y_i^t \qquad i\in N, t\in\mathcal{T}$$

$$x_{ij}^t\in\{0,1\} \qquad ((i,t)(j,t+\tau_{ij}))\in\mathcal{A}^{\mathcal{T}}.$$

Constraints (3.2a) and (3.2b) ensure vehicle flow balance and ensure that all m vehicles are returned to the depot at the end of the planning horizon. Constraints (3.2c) together with the requirement that each x_{ij}^t variable is binary ensure that at most one vehicle can be visiting a customer at any one time. Constraints (3.2d) ensure product flow balance and enforce that product arriving in a vehicle at a customer is either delivered at that customer or remains on the vehicle. Constraints (3.2e) link the product flows to the vehicle flows. Constraints (3.2f) and (3.2g) model product usage at a customer and inventory balance. Constraints (3.2h) ensure that inventory at a customer is sufficient, after each delivery, to meet the customer demand in the coming period and never exceeds the local storage capacity. When all travel times

are strictly positive, the time-expanded network is acyclic. As a consequence, there is no need to explicitly forbid subtours in the model, and the given vehicle flow balance constraints ensure that the number of vehicles that are away from the depot at any time does not exceed m. In the case that some travel times are zero, the situation is more complicated: we discuss this further in Section 3.5. The relationship between the model presented above and the model presented by [2] is discussed in Appendix A.

This model has a nice structure, in the sense that for fixed x, the model is a network flow model. As a consequence, when all data is integer, and the problem is feasible, there must exist a solution in which all variables take on integer values.

Proposition 12. *For fixed x, the above model* (3.2) *in the w, y and z variables takes the form of a network flow problem.*

Proof. For fixed x, the w, y and z variables in any feasible solution to the above model are those that satisfy the constraints

$$\sum_{j \in N} w_{ji}^{t-\tau_{ji}} - \sum_{j \in N} w_{ij}^{t} = y_i^t \quad i \in N, t \in \mathcal{T} \tag{3.3a}$$

$$0 \leq w_{ij}^t \leq Q x_{ij}^t \qquad i \in N_0, t \in \mathcal{T}, j \in \delta^+(i,t) \tag{3.3b}$$

$$z_i^t = z_i^{t-1} + y_i^t - u_i \qquad i \in N, t \in \mathcal{T} \setminus \{0\} \tag{3.3c}$$

$$z_i^0 = I_i^0 + y_i^0 - u_i, \qquad i \in N \tag{3.3d}$$

$$u_i \leq z_i^t \leq C_i \qquad i \in N, t \in \mathcal{T} \tag{3.3e}$$

$$0 \leq y_i^t \qquad i \in N, t \in \mathcal{T} \tag{3.3f}$$

These constraints can be shown to define a network flow polyhedron. The network has two nodes for each pair (i,t) with $i \in N_0$ and $t \in \mathcal{T}$. Let n_i^t denote the first and m_i^t denote the second such node. First nodes are linked by arcs in $N_0 \times N_0$, so there is a subnetwork with arcs of the form $(n_i^t, n_j^{t+\tau_{ij}})$ for $(i,j) \in N_0 \times N_0$, with capacity range

$[0, Qx_{ij}^t]$ and flow on the arc given by the variable w_{ij}^t. The flow capacity constraints on these arcs are thus precisely constraints (3.3b). The first and second nodes for each pair (i,t) are linked by arcs of the form (n_i^t, m_i^t), with flow lower bound zero, carrying flow given by variable y_i^t. First nodes are required to be transshipment nodes (have zero net outflow), so the flow conservation equation at nodes n_i^t are precisely the equations (3.3a). There are also arcs between the second nodes, of the form (m_i^t, m_i^{t+1}), with capacity range $[u_i, C_i]$, carrying flow given by variable z_i^t. Thus the arc capacity constraints are precisely the constraints (3.3e). The nodes of the form m_i^t are required to have net inflow of u_i, if $t \in \mathcal{T} \setminus \{0\}$ and $u_i - I_i^0$ if $t = 0$. Thus the flow conservation constraints at second nodes of the form m_i^t are precisely (3.3c) and (3.3d). With the addition of appropriate dummy nodes and arcs to balance the flow, the constraints above clearly define a network flow polyhedron. □

Unfortunately, the time-expanded network formulation (3.2) may be prohibitively large. Furthermore, while a *correct* discretization parameter, Δ, which guarantees that an optimal solution to (3.2) gives an optimal solution to the continuous time problem, must exist in theory, its value is, in practice, unknown. However, by selecting a (possibly incorrect) value of Δ, *a priori*, and adjusting time related parameters carefully, we can construct smaller, more manageable MIP formulations that provide either a lower or an upper bound on the optimal value of the original formulation. For example, if $H = 24$ and $\Delta = 2$, then the resulting formulation uses times $\mathcal{T} = \{0, 1, 2, \ldots, 11\}$, stated in periods of length 2, and if $\Delta = 4$, the resulting formulation uses times $\mathcal{T} = \{0, 1, 2, \ldots, 5\}$, stated in periods of length 4. When H is not divisible by Δ, then we may use $\mathcal{T} = \{0, 1, \ldots, \lfloor \frac{H}{\Delta} \rfloor\}$. When using a given time discretization, the time related parameters need to be adjusted. For example, the usage rate $\hat{u}_i$ at customer i has to be adjusted to $\Delta\hat{u}_i$, i.e., in each time interval of length Δ, $\Delta\hat{u}_i$ units of product are consumed. Adjusting the travel time is more involved as it requires making choices. The two natural choices are $\lfloor \hat{\tau}_{ij}/\Delta \rfloor$, which implies that the

travel time may be decreased, and $\lceil \hat{\tau}_{ij}/\Delta \rceil$, which implies that the travel time may be increased. Depending on this choice, the resulting formulation produces either a lower or an upper bound. The former is described in detail in Section 3.5.2 and the latter in Section 3.6.1.

Before doing so, we introduce some notation that will be useful in the remainder of the chapter. Recall that a route, r, specifies a sequence of customer deliveries, starting and ending at the depot. If r specifies k deliveries to customers $i_j \in N$, $j = 1, \ldots, k$, in the sequence $i_1, \ldots, i_k$, the cost of the route, which we denote by c^r, is given by $c^r = \sum_{j=0}^{k} c_{i_j, i_{j+1}}$, where $i_0 = i_{k+1} = 0$. We will sometimes write $i \in r$ to indicate customer $i \in N$ is in route r. A route also specifies quantities delivered, say q_j^r is the quantity delivered to customer i_j at the jth delivery in route r, for $j = 1, \ldots, k$. For the route to be feasible, it must be that $\sum_{j=1}^{k} q_j^r \leq Q$. Recalling that there may be more than one delivery to a customer in the same route, when the context ensures there is no chance of confusion, we also use $q_i^r = \sum_{j \in \{1,\ldots,k\}:\, i_j = i} q_j^r$ to denote the total quantity delivered to customer $i \in r$ on route r. Naturally, for r a feasible route, $\sum_{i \in r} q_i^r \leq Q$ also.

3.5 Lower bounds

We first describe a simple lower bound that can be calculated without the need to solve an integer program, for instances in which the costs are symmetric and satisfy the triangle inequality.

3.5.1 A simple lower bound

Proposition 13. *Provided the costs $(c_{ij})_{i,j \in N_0}$ are symmetric and satisfy the triangle inequality, a lower bound on the optimal value of the CIRP is given by*

$$2 \sum_{i \in N} \left(\frac{H\hat{u}_i - I_i^0}{Q} \right) c_{0i}.$$

The proof relies on the following lemma.

Lemma 4. *Assume that the costs are symmetric and satisfy the triangle inequality, and consider a route r. Then for any $\lambda \geq 0$ such that $\sum_{i \in r} \lambda_i \leq 1$, it must be that*

$$c^r \geq 2 \sum_{i \in r} \lambda_i c_{0i}.$$

Proof. For any $\lambda \geq 0$ with $\sum_{i \in r} \lambda_i \leq 1$, it must be that $\sum_{i \in r} \lambda_i c_{0i} \leq \max_{i \in r}\{c_{0i}\}$. Since the costs are symmetric, twice the latter value gives the cost of visiting the customer in r that is farthest from the depot. Since the costs satisfy the triangle inequality, visiting one customer of the route is cheaper than visiting all of them. It follows that

$$c^r \geq 2 \max_{i \in r}\{c_{0i}\} \geq 2 \sum_{i \in r} \lambda_i c_{0i}$$

as required. □

Proof of Proposition 13. During the planning horizon, customer $i \in N$ consumes $\hat{u}_i H$ units of product. Therefore, the amount delivered on the routes visiting customer i during the planning horizon needs to be at least $\hat{u}_i H - I_i^0$. Let R be the set of routes in an optimal solution and let q_i^r be the quantity delivered to customer $i \in N$ on route $r \in R$. Thus, $\sum_{r \in R:\, i \in r} q_i^r \geq \hat{u}_i H - I_i^0$ for all $i \in N$.

Next, we apply Lemma 4 using $\lambda_i = \frac{q_i^r}{Q}$ for customer $i \in r$ with $r \in R$. Since r is feasible, $\sum_{i \in r} q_r^i \leq Q$ so $\sum_{i \in r} \lambda_i \leq 1$. This implies that the optimal value of the

CIRP, given by the sum of the costs of all routes, satisfies

$$\begin{aligned}\sum_{r\in R} c^r &\geq \sum_{r\in R} 2\sum_{i\in r}\frac{q_i^r}{Q}c_{0i}\\ &= 2\sum_{i\in N}\frac{c_{0i}}{Q}\sum_{r\in R:\, i\in r} q_i^r\\ &\geq 2\sum_{i\in N}\frac{c_{0i}}{Q}(\hat{u}_i H - I_i^0),\end{aligned}$$

which completes the proof. □

While this bound is very easy to calculate, we found that it is quite weak in practice. Hence, stronger lower bounds are required. The next section discusses an approach that, for more computational effort, can yield much stronger bounds.

3.5.2 A lower bound integer programming model

Our lower bound model is based on (3.2), in the sense that it uses the same variables and parameter names for a given discretization parameter, Δ. Specifically, $T = \lceil H/\Delta \rceil$, $u_i = \hat{u}_i\Delta$ for all $i \in N$, and $\tau_{ij} = \lfloor \hat{\tau}_{ij}/\Delta \rfloor$ for all $i, j \in N_0$, $i \neq j$. It is important to note that this may introduce travel times of length zero. We again take $\tau_{ii} = 1$ for all $i \in N_0$. To obtain a formulation that yields a lower bound on the optimal value of the continuous time problem, (3.2) must be modified. The modifications required are:

- the constraints $u_i \leq z_i^t \leq C_i$ in (3.2h) must be relaxed to

$$u_i \leq z_i^t \leq C_i + u_i, \qquad i \in N, t = 0, \ldots, T-2 \tag{3.4a}$$

$$(H/\Delta - (T-1))u_i \leq z_i^{T-1} \leq C_i + (H/\Delta - (T-1))u_i, \quad i \in N, \tag{3.4b}$$

where $(H/\Delta - (T-1))u_i$ is the number of units of product consumed by customer i in the part of the planning horizon from $\Delta(T-1)$ to H, and

- more than one vehicle at a customer at a time must be allowed, which can be effected by removing constraints (3.2c) and allowing each x_{ij}^t variable to be a non-negative integer, not necessarily binary.

Note that if Δ divides evenly into H, then $T = \lceil H/\Delta \rceil = H/\Delta$ and so $H/\Delta - (T-1) = 1$. Otherwise $0 \leq H/\Delta - (T-1) < 1$ and so the first modification is indeed a relaxation.

We call the resulting model the *Lower Bound Model (LBM)*.

3.5.3 Properties of the Lower Bound Model

We prove that the LBM indeed yields a lower bound on the value of the original, continuous time, problem (CIRP) by showing that any feasible solution for a continuous time instance can be mapped to a feasible solution for LBM having the same cost.

Proposition 14. *The optimal value of the Lower Bound Model is a lower bound on the optimal value of the CIRP.*

Proof. We will prove that any solution for the CIRP can be transformed into a feasible solution for LBM, without any additional cost. First, we recall that a CIRP solution consists of a finite set of deliveries to each customer, at a finite set of time points during the planning horizon, delivered by the m vehicles, each undertaking a sequence of routes, each of which starts and ends at the depot. The transformation "shifts" all deliveries made in the time interval $[\Delta t, \Delta(t+1))$ in the CIRP solution to occur at LBM time index $t \in \mathcal{T}$. Note that for any $s \in [0, H)$ it must be that $\lfloor s/\Delta \rfloor \in \mathcal{T}$. Now observe that if, in the CIRP solution, any vehicle moves from i to j departing at time $s \in [0, H)$, where $i, j \in N_0$, $i \neq j$, the time-expanded arc $((i, \lfloor s/\Delta \rfloor), (j, \lfloor s/\Delta \rfloor + \tau_{ij}))$ must exist in $\mathcal{A}^{\mathcal{T}}$. This holds since $s + \hat{\tau}_{ij} \in [0, H]$ and

$$\lfloor s/\Delta \rfloor + \tau_{ij} = \lfloor s/\Delta \rfloor + \lfloor \hat{\tau}_{ij}/\Delta \rfloor \leq \lfloor (s + \hat{\tau}_{ij})/\Delta \rfloor \in \mathcal{T}$$

provided $s + \hat{\tau}_{ij} < H$. Note that it may be that $s + \hat{\tau}_{ij} = H$, in which case it must

be that the vehicle is returning to the depot, so $j = 0$. In this case, $\lfloor {}^{s}/_{\Delta} \rfloor + \tau_{ij} \leq \lfloor {}^{H}/_{\Delta} \rfloor \in \{T-1, T\}$ and certainly $((i, \lfloor {}^{s}/_{\Delta} \rfloor), (0, \lfloor {}^{s}/_{\Delta} \rfloor + \tau_{ij})) \in \mathcal{A}^{\mathcal{T}}$.

The transformation takes the variable x_{ij}^t in the LBM model to be the number of vehicles traveling from i to j departing at any time in $[\Delta t, \Delta(t+1))$ in the CIRP solution. Similarly, the w_{ij}^t variable in the LBM model will be taken to be the total units of product carried on any vehicle traveling from i to j, departing at any time in $[\Delta t, \Delta(t+1))$, in the CIRP solution. Feasibility of the CIRP solution ensures that LBM constraints (3.2a), (3.2b) and (3.2e) are satisfied, and it is now clear why, due to aggregation of multiple vehicle movements in the construction of the x_{ij}^t variables, these are relaxed in the LBM to allow non-binary integers and (3.2c) is omitted.

This construction of the x_{ij}^t variables ensures that the cost of the LBM solution is identical to the cost of the CIRP solution: each movement of a vehicle from i to j in the latter adds one to some x_{ij}^t variable, adding c_{ij} to the LBM objective function.

It remains to show that the customer inventory variables in the LBM can be set correctly. We first use the deliveries and inventory at each customer over the horizon $[0, H]$ in the CIRP solution to set the y_i^t and z_i^t variables in the LMB and show these are feasible in the LBM constraints (3.2f), (3.2g), (3.4a) and (3.4b). (Recall that (3.4a) and (3.4b) replace (3.2h) in the LBM.) We then explain why they are consistent with the vehicle routing variables defined above, ensuring (3.2d).

Given a CIRP solution, let J_i be a finite index set for the set of deliveries to customer $i \in N$, let $v_j^i \in [0, H]$ denote the time at which delivery $j \in J_i$ is made to customer i and let η_j^i denote the number of units of product delivered to customer i at this time.

From this data, the function $\hat{z}_i(s)$, representing the inventory of customer i at time Δs in the CIRP solution for each $s \in [0, {}^{H}/_{\Delta}]$, can readily be constructed. (We define $\hat{z}_i(s)$ to be the inventory at time Δs *excluding* any deliveries made at precisely this time.) Note that for feasibility of the CIRP solution, it must be that $0 \leq \hat{z}_i(s) \leq C_i$

for all $s \in [0, {}^{H}/_{\Delta}]$. Since the LBM "rounds down" travel times, to create a feasible solution for the LBM from the CIRP solution, we "shift" all deliveries made in the time interval $[\Delta t, \Delta(t+1))$ to occur at LBM time index $t \in \mathcal{T}$. It is thus helpful to define the index set of deliveries made in this interval: $J_i(t) = \{j \in J_i : \lfloor {}^{v_j^i}/_{\Delta} \rfloor = t\}$ for each $t \in \mathcal{T}$, $i \in N$. Thus in the CIRP solution, it must be that

$$\hat{z}_i(t) = \hat{z}_i(t-1) + \sum_{j \in J_i(t-1)} \eta_j^i - u_i, \qquad \forall t \in \mathcal{T} \setminus \{0\}, i \in N, \tag{3.5}$$

and

$$\hat{z}_i(0) = I_i^0, \qquad \forall i \in N. \tag{3.6}$$

Also, since the CIRP solution is feasible,

$$0 \leq \hat{z}_i(t) \leq C_i, \qquad \forall t \in \mathcal{T}, i \in N. \tag{3.7}$$

and, since $0 \leq \hat{z}_i({}^{H}/_{\Delta}) \leq C_i$, it must be that

$$0 \leq \hat{z}_i(T-1) + \sum_{j \in J_i(T-1)} \eta_j^i - ({}^{H}/_{\Delta} - (T-1))u_i \leq C_i, \qquad \forall i \in N. \tag{3.8}$$

Construction of the LBM solution is completed by setting $y_i^t = \sum_{j \in J_i(t)} \eta_j^i$ and $z_i^t = \hat{z}_i(t) + y_i^t$ for each $t \in \mathcal{T}$, $i \in N$. Since $J_i(t)$ includes the index for any delivery made precisely *at* time Δt in the CIRP solution, we see that z_i^t is the inventory after any deliveries made at $t \in \mathcal{T}$ in the LBM solution. Clearly, for each $t \in \mathcal{T} \setminus \{0\}$,

$i \in N$, we have, by (3.5), that

$$\begin{aligned} z_i^t &= \hat{z}_i(t) + y_i^t \\ &= \hat{z}_i(t-1) + \sum_{j \in J_i(t-1)} \eta_j^i - u_i + y_i^t \\ &= \hat{z}_i(t-1) + y_i^{t-1} - u_i + y_i^t \\ &= z_i^{t-1} - u_i + y_i^t, \end{aligned}$$

ensuring (3.2f) holds. Also, for each $i \in N$, we have, by (3.6), that

$$z_i^0 = \hat{z}_i(0) + y_i^0 = I_i^0 + y_i^0,$$

ensuring (3.2g) holds. Now for all $i \in N$ and $t = 0, 1, \ldots, T-2$, we have that

$$z_i^t = \hat{z}_i(t) + y_i^t = \hat{z}_i(t) + \sum_{j \in J_i(t)} \eta_j^i = \hat{z}_i(t+1) + u_i,$$

by (3.5), and so $\hat{z}_i(t+1) = z_i^t - u_i$. Hence, by (3.7), it must be that for all $i \in N$ and $t = 0, 1, \ldots, T-2$,

$$\begin{aligned} 0 \leq \hat{z}_i(t+1) \leq C_i \qquad &\iff \qquad 0 \leq z_i^t - u_i \leq C_i \\ &\iff \qquad u_i \leq z_i^t \leq C_i + u_i, \end{aligned}$$

ensuring (3.4a) holds. Finally, for all $i \in N$,

$$z_i^{T-1} = \hat{z}_i(T-1) + y_i^{T-1} = \hat{z}_i(T-1) + \sum_{j \in J_i(T-1)} \eta_j^i,$$

and so by (3.8), it must be that

$$0 \leq z_i^{T-1} - ({}^{H}/_{\Delta} - (T-1))u_i \leq C_i$$

$$\iff$$

$$({}^{H}/_{\Delta} - (T-1))u_i \leq z_i^{T-1} \leq C_i + ({}^{H}/_{\Delta} - (T-1))u_i,$$

ensuring that (3.4b) holds.

To complete construction of the LBM feasible solution, we need to be sure that for any vehicle route making a delivery to customer i at time $s \in [0, H]$ in the CIRP solution, there is a corresponding route feasible to the LBM that delivers at time index $\lfloor {}^{s}/_{\Delta} \rfloor$. To see that this must be so, consider two consecutive customer deliveries in a CIRP vehicle route, or a departure from the depot followed by a customer delivery, or a customer delivery followed by departure from the depot. Suppose these two events occur at times $s_1, s_2 \in [0, H]$ with $s_1 \leq s_2$, and at locations $i_1, i_2 \in N_0$, respectively. Since the CIRP solution is feasible, we have that $s_1 + \hat{\tau}_{i_1 i_2} \leq s_2$ in the case $i_1 \neq i_2$ and $s_1 \leq s_2$ otherwise. In the former case, we get

$$s_1 + \hat{\tau}_{i_1 i_2} \leq s_2 \iff {}^{s_1}/_{\Delta} + {}^{\hat{\tau}_{i_1 i_2}}/_{\Delta} \leq {}^{s_2}/_{\Delta}$$

$$\implies \lfloor {}^{s_1}/_{\Delta} \rfloor + \lfloor {}^{\hat{\tau}_{i_1 i_2}}/_{\Delta} \rfloor \leq \lfloor {}^{s_1}/_{\Delta} + {}^{\hat{\tau}_{i_1 i_2}}/_{\Delta} \rfloor \leq \lfloor {}^{s_2}/_{\Delta} \rfloor$$

and hence $\lfloor {}^{s_1}/_{\Delta} \rfloor + \tau_{i_1 i_2} \leq \lfloor {}^{s_2}/_{\Delta} \rfloor$. Thus the two events can occur consecutively in a LBM feasible route, at $\lfloor {}^{s_1}/_{\Delta} \rfloor, \lfloor {}^{s_2}/_{\Delta} \rfloor \in \mathcal{T}$, respectively. If $i_1 = i_2$, then obviously $s_1 \leq s_2$ implies $\lfloor {}^{s_1}/_{\Delta} \rfloor \leq \lfloor {}^{s_2}/_{\Delta} \rfloor$, so the two events can also occur consecutively in an LBM feasible route. □

Having established that the LBM deserves its name, it is natural to ask whether or not its value is guaranteed to approach the CIRP value as Δ decreases towards zero. Unfortunately, the omission of (3.2c) from the LBM means that it may not. The

LBM solution can have two vehicles visiting a customer at the same time, in order to move product from one vehicle to the other; doing so can reduce costs. This is a well known issue in split delivery vehicle routing problems, which makes them challenging to model. In split delivery routing, this issue is often resolved by a vehicle indexed formulation. The same device may be used here: there is a natural alternative form of the LBM based on routing variables indexed by vehicle. In practice, we found that the potential bound improvement from use of a vehicle indexed version of the LBM did not outweigh the extra computing time this much larger formulation required: even for small instances, solving the vehicle indexed formulation exactly was not possible in moderate time, and its best bound when stopped early did not match that of the LBM above, computed in the same time.

3.5.4 Strengthening the Lower Bound Model

Eliminating Depot Subtours

When Δ is relatively large, the LBM has another weakness. It occurs when for one or more pairs $(i, j) \in N_0 \times N_0$, we have $\tau_{ij} = \lfloor \hat{\tau}_{ij}/\Delta \rfloor = 0$, which induce cycles in the time-expanded network (e.g., a cycle from i to j and back to i that takes up no time). The LBM allows a vehicle to traverse such a zero travel time cycle even if it is disconnected from the vehicle origin node, $(0, 0)$. For some $t \in \mathcal{T}$ and some set $S \subseteq N_0$ with $\tau_{ij} = 0$ for all distinct $i, j \in S$, the x_{ij}^t variables may induce a cycle even if $x_{hi}^{t-\tau_{hi}} = 0$ for all $i \in S$ and all $h \in \delta^-(i, t) \setminus S$. This violates the property of any CIRP feasible solution that, when transformed to an LBM solution with vehicle route variables x, the subgraph in $(\mathcal{N}^{\mathcal{T}}, \mathcal{A}^{\mathcal{T}})$ induced by x decomposes into paths (not necessarily simple) from $(0, 0)$ to $(0, T)$. In the presence of a zero travel time cycle, the LBM solution may violate this property. However, if the cycle does *not* include the depot, then the product flow variables, w_{ij}^t and the constraints (3.2d) and (3.2e) prevent the cycle from being used to deliver product to customers, so there is no incentive for such

a cycle to appear in the LBM solution. On the other hand, if the cycle *does* include the depot, it can be used to deliver product to customers, without being part of a path connecting the origin node, $(0, 0)$, to the destination, $(0, T)$. Hence the vehicle traversing the cycle is omitted from the vehicle count constraint (3.2b). We call such a cycle in the LBM solution a *depot subtour* and illustrate its occurrence in Figure 3.4. The example shown in Figure 3.4 has two customers, $I_1^0 = 2$, $C_1 = 4$, $I_2^0 = 1$, $C_2 = 3$, $\hat{u}_i = 1$ for $i = 1, 2$, $c_{ij} = \hat{\tau}_{ij} = 1$ for all $i, j \in \{0, 1, 2\}, i \neq j$, a single vehicle, $Q = 6$, and planning horizon $H = 6$. An optimal CIRP solution is for the vehicle to use the time-expanded node sequence $(0, 0), (2, 1), (1, 2), (0, 3), (2, 4), (1, 5), (0, 6)$, visiting customer 2 then customer 1, returning to the depot, and repeating (shown on the left in Figure 3.4). On the first route, the vehicle delivers 3 units to both customers, and on the second route, delivers at least 2 units to customer 2 and at least 1 unit to customer 1. The cost is 6. If we take $\Delta = 2$ and solve the LBM, the optimal solution uses the time-expanded node sequence $(0, 0), (2, 0), (2, 1), (2, 2), (0, 2), (0, 3)$ and $(0, 1), (1, 1), (0, 1)$, the latter being a depot subtour (shown on the right in Figure 3.4). On the first sequence the vehicle delivers a total of 5 units to customer 2, delivering at least 1 unit on arrival. For example, it may deliver 2 units at time index 0, 2 units at time index 1 and the remaining 1 unit at time index 2. On the depot subtour, it delivers 4 units to customer 1. This meets customer demand with a cost of only 4. However, in the second period, at time index 1, the vehicle is apparently in two places at once: a phantom vehicle has been used on a depot subtour.

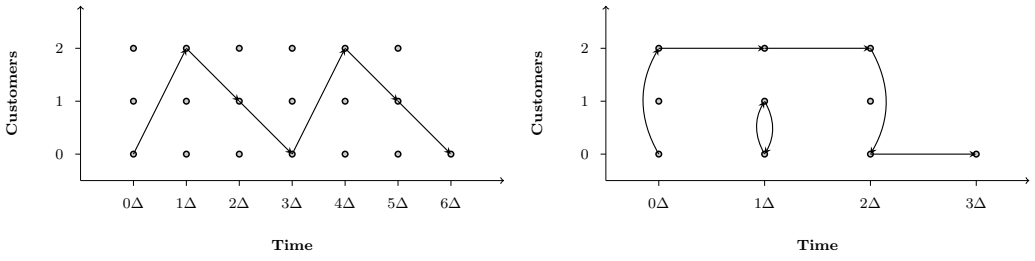

Figure 3.4: An example of a depot subtour.

Depot subtours can be avoided by the use of a vehicle indexed formulation. However, as discussed earlier, such a formulation is not as effective in practice. Alternatively, depot subtours can be avoided by ensuring that Δ is small enough that no zero travel time cycles including the depot occur in the time-expanded network. However, this requirement may make it challenging to manage the size of the MIP formulation. Instead, we introduce auxiliary variables and constraints to eliminate depot subtours. The variable $\tilde{w}_{ij}^t$ represents integer flow of a commodity that must be transported from node $(0,0)$ to node $(0,T)$, with at least one unit of flow carried by every vehicle on every arc, and one unit of flow delivered to each location in N_0 per vehicle entering the location at any time $t \in \{1,\ldots,T-1\}$. The following *depot subtour elimination constraints* are added to the LBM model:

$$\sum_{j\in\delta_t^-(i)} \tilde{w}_{ji}^{t-\tau_{ij}} - \sum_{j\in\delta_t^+(i)} \tilde{w}_{ij}^t = \sum_{j\in\delta_t^-(i)} x_{ji}^{t-\tau_{ji}} \quad \forall (i,t) \in \mathcal{N}^{\mathcal{T}} \setminus \{(0,0),(0,T)\} \tag{3.9a}$$

$$x_{ij}^t \le \tilde{w}_{ij}^t \le M_{ij}^t x_{ij}^t \qquad \forall ((i,t),(j,t+\tau_{ij})) \in \mathcal{A}^{\mathcal{T}}, t+\tau_{ij} \neq T, \tag{3.9b}$$

where M_{ij}^t is a large number. These constraints ensure that the x variables induce a subgraph in $(\mathcal{N}^{\mathcal{T}}, \mathcal{A}^{\mathcal{T}})$ that can be decomposed into (possibly non-elementary) paths from $(0,0)$ to $(0,T)$. In the example, we see that constraints (3.9a) require that $\tilde{w}_{01}^1 - \tilde{w}_{10}^1 = 1$ and $\tilde{w}_{10}^1 - \tilde{w}_{01}^1 = 1$, which is impossible. Thus the depot subtour $(0,1),(1,1),(0,1)$ is eliminated.

Determining a valid choice for M_{ij}^t is not easy. It needs to be an upper bound on the number of times a vehicle can arrive at customer i at some time point t' in the set $\{t+\tau_{ij},\ldots,T-1\}$. In Appendix B, we suggest one approach to calculating a valid choice.

Valid Inequalities

We adapt several valid inequalities from the period inventory routing problem and suggest one additional class.

Period inventory routing problems take a time discretization as part of the problem description, with consumption of each customer in each time period a given parameter. Vehicles start and end routes within a single time period, and it is assumed that only the inventory at the end of each period, after any deliveries have been added and demand subtracted, must be nonnegative and no greater than the customer's storage capacity. In this context, [71] present several valid inequalities. We adapt three classes of valid inequality they present to our setting. In particular, we adapt inequalities (18) [Theorem 2], (20) [Theorem 4] and (22) [Theorem 6].

First, a lower bound on the total number of visits to a customer over the planning horizon is exploited. In the CIRP, we observe that customer $i \in N$ requires $\hat{u}H - I_i^0$ units of product in total to be delivered during the time, and thus requires at least $\left\lceil \frac{\hat{u}_i H - I_i^0}{Q} \right\rceil$ vehicle visits with an intervening return to the depot. Thus the inequalities

$$\left\lceil \frac{\hat{u}_i H - I_i^0}{Q} \right\rceil \leq \sum_{t \in \mathcal{T}} \sum_{j \in \delta^-(i,t) \setminus \{i\}} x_{ji}^{t-\tau_{ji}}, \qquad i \in N. \tag{3.10}$$

must be valid for the LBM. Note that the right-hand side of this constraint excludes vehicles waiting at customer i, since a waiting vehicle will not have had an intervening return to the depot. In the case that $C_i < Q$, a stronger lower bound on the number of visits to customer $i \in N$ is $\left\lceil \frac{\hat{u}_i H - I_i^0}{C_i} \right\rceil$, however in this case waiting vehicles must be counted. Thus, in this case, we obtain another class of inequalities,

$$\left\lceil \frac{\hat{u}_i H - I_i^0}{C_i} \right\rceil \leq \sum_{t \in \mathcal{T}} \sum_{j \in \delta^-(i,t)} x_{ji}^{t-\tau_{ji}}, \qquad i \in N \text{ with } C_i < Q. \tag{3.11}$$

When $C_i < Q$, neither inequality from the class (3.10) or (3.11) may dominated the

other; both may be useful.

[67] present inequalities similar to (3.10) and (3.11), but derive a lower bound on the number of visits to a customer that must occur in a given time interval. In our setting, their inequalities correspond to

$$\left\lceil \frac{u_i(t_2 - t_1 + 1) - C_i}{Q} \right\rceil \leq \sum_{t=t_1+1}^{t_2} \sum_{j \in \delta^-(i,t) \setminus \{i\}} x_{ji}^{t-\tau_{ji}}, \qquad i \in N, 0 \leq t_1 < t_2 \leq T-1, \tag{3.12}$$

and,

$$\left\lceil \frac{u_i(t_2 - t_1 + 1) - C_i}{C_i} \right\rceil \leq \sum_{t=t_1+1}^{t_2} \sum_{j \in \delta^-(i,t)} x_{ji}^{t-\tau_{ji}}, \qquad i \in N, 0 \leq t_1 < t_2 \leq T-1 \text{ with } C_i < Q, \tag{3.13}$$

where to be valid for LBM, we have to replace C_i with $C_i + u_i$ for $i \in N$.

The observation that the inventory on hand at the start of a period must be sufficient to sustain the customer until its next delivery can also be exploited. This needs to be done carefully in the LBM model, since the deliveries in a CIRP solution during the interval starting at time $t\Delta$ are "mapped to" deliveries at time point $t \in \mathcal{T}$. Thus, if there are no deliveries to customer i at any of time points $t_1+1, t_1+2, \ldots, t_2$, in the LBM, the inventory after any delivery at t_1 must be sufficient to meet demand in the intervals starting at times $t_1\Delta, (t_1+1)\Delta, \ldots, t_2\Delta$, i.e., for a time duration of $(t_2 - t_1 + 1)\Delta$. Thus inventory at i must be at least $\hat{u}_i(t_2 - t_1 + 1)\Delta = u_i(t_2 - t_1 + 1)$, and we have the valid inequality

$$z_i^{t_1} \geq u_i(t_2 - t_1 + 1) \left(1 - \sum_{t=t_1+1}^{t_2} \sum_{j \in \delta^-(i,t)} x_{ji}^{t-\tau_{ji}} \right), \qquad i \in N, 0 \leq t_1 < t_2 \leq T-1. \tag{3.14}$$

Using ideas from [67], we can strengthen (3.14) to

$$z_i^{t_1} \geq u_i(t_2 - t_1 + 1)\left(1 - \min\left\{\frac{\min\{C_i, Q\}}{u_i(t_2 - t_2 + 1)}, 1\right\} \sum_{t=t_1+1}^{t_2} \sum_{j \in \delta^-(i,t)} x_{ji}^{t-\tau_{ji}}\right), \qquad i \in N, 0 \leq t_1 < t_2 \leq T-1, \tag{3.15}$$

where, again, we have to replace C_i with $C_i + u_i$ for $i \in N$ to be valid for LBM.

Finally, we make use of the observation that whenever a vehicle arrives at a customer it must have departed the depot at some time sufficiently beforehand. To use this observation, we must take care concerning the triangle inequality for travel times. In the context of our LBM, we note that, even though the original travel times, $\hat{\tau}$, are assumed to satisfy the triangle inequality, the travel times scaled to conform to a discretization with parameter Δ may not. For example, consider the case that $\hat{\tau}_{ij} = \hat{\tau}_{jk} = 3.2$ and $\hat{\tau}_{ik} = 6$, for three distinct customers i, j, k. The triangle inequality is satisfied here, as $\hat{\tau}_{ij} + \hat{\tau}_{jk} = 3.2 + 3.2 = 6.4 \geq 6 = \hat{\tau}_{ik}$. However, if we take $\Delta = 2$, then

$$\begin{aligned} t_{ij} + t_{jk} &= \lfloor \hat{\tau}_{ij}/\Delta \rfloor + \lfloor \hat{\tau}_{jk}/\Delta \rfloor = \lfloor 3.2/2 \rfloor + \lfloor 3.2/2 \rfloor \\ &= \lfloor 1.6 \rfloor + \lfloor 1.6 \rfloor = 1 + 1 = 2 \not\geq 3 = \lfloor 6/2 \rfloor \\ &= \lfloor \hat{\tau}_{ik}/\Delta \rfloor = t_{ik}. \end{aligned}$$

Thus we must adapt the statement of inequality to account for this. Letting $\underline{\tau}_{0i}$ be the length of the shortest path from 0 to i in the complete network on nodes N_0, with length of arc (j, k) taken to be τ_{jk}, we have that the class of constraints

$$\sum_{j \in \delta^-(i,t)} x_{ji}^{t-\tau_{ji}} \leq \sum_{s=0}^{t-\underline{\tau}_{0i}} \sum_{j \in \delta^+(0,s), j \neq 0} x_{0j}^{s}, \qquad (i,t) \in \mathcal{N}^{\mathcal{T}} \tag{3.16}$$

is valid for the LBM.

3.6 Constructing feasible delivery plans

Next, we investigate optimization models that may produce feasible solutions to the continuous time problem, CIRP. We consider two alternative approaches. The first is to develop a different, upper bound, model, based on (3.2) with appropriate parameter choices and modifications. The second is based on the solution to the LBM, and attempts to adjust it to obtain a feasible CIRP solution.

3.6.1 An upper bound integer programming model

Like the LBM, our upper bound model is based on (3.2), using the same variables and parameter names for a give discretization parameter, Δ. As for the LBM, $T = \lceil H/\Delta \rceil$ and $u_i = \hat{u}_i \Delta$ for all $i \in N$, but the travel times are now rounded up, rather than down: $\tau_{ij} = \lceil \hat{\tau}_{ij}/\Delta \rceil$ for all $i, j \in N_0$, $i \neq j$. To obtain a formulation that yields an upper bound on the optimal value of the continuous time problem, (3.2) must be modified. The modifications required are:

- the constraints $u_i \leq z_i^t \leq C_i$ in (3.2h), for the case $t = T - 1$, must be replaced by

$$({}^{H}/_{\Delta} - (T-1))u_i \leq z_i^{T-1} \leq C_i, \qquad \forall i \in N, \tag{3.17}$$

  where $({}^{H}/_{\Delta} - (T-1))u_i$ is the number of units of product consumed by customer i in the part of the planning horizon from $(T-1)\Delta$ to H, and

- some arcs must be removed, according to

$$\mathcal{A}^{\mathcal{T}} := \mathcal{A}^{\mathcal{T}} \setminus \{((i,t),(0,T)) : t\Delta + \hat{\tau}_{i0} > H\}. \tag{3.18}$$

Both are modifications to (3.2) only if Δ does not divide evenly into H. In this case, for the final time interval induced by the discretization, $[(T-1)\Delta, T\Delta]$, with $T\Delta > H$, the inventory on hand at each customer at the start of the interval only needs to be

enough to supply the customer up to time H. Also, all vehicles must return to the depot by time H, which is modeled by the sink node $(0, T)$ in the discretized problem, and so arc $((i, t), (0, T)) \in \mathcal{A}^{\mathcal{T}}$ should only be used if $\Delta t + \hat{\tau}_{i0} \leq H$.

We call the resulting model the *Upper Bound Model (UBM)*. That any feasible solution to the UBM is also a feasible solution to the CIRP is quite obvious. A vehicle visit to customer i (or the depot) at time point $t \in \mathcal{T}$ followed by a visit to customer j (or the depot) at time point $t+\tau_{ij} \in \mathcal{T}$ in the UBM corresponds to a vehicle visit to customer i at time $t\Delta \in [0, H]$ followed by travel to j, arriving at time $t\Delta + \hat{\tau}_{ij}$, and waiting at j for time $\tau_{ij}\Delta - \hat{\tau}_{ij} > 0$, by the definition of τ_{ij}. The inventory constraints ensure that the customer deliveries made at times $t\Delta$ for $t \in \mathcal{T}$ are sufficient to meet the customer demand over the whole planning period $[0, H]$. The proposition below follows.

Proposition 15. *If the Upper Bound Model is feasible, then it provides a feasible solution for the CIRP.*

We note that constraints similar to those for strengthening the LBM may be used to strengthen the UBM. Specifically, (3.10)–(3.13), and (3.15) may all be used as stated, while, since the triangle inequality for $\hat{\tau}$ implies the triangle inequality for τ in the UBM, (3.16) is applied simply as

$$\sum_{j \in \delta^-(i,t)} x_{ji}^{t-\tau_{ji}} \leq \sum_{s=0}^{t-\tau_{0i}} \sum_{j \in \delta^+(0,s)} x_{0j}^{s}, \qquad (i, t) \in \mathcal{N}^{\mathcal{T}}. \tag{3.19}$$

3.6.2 Converting solutions to LBM

In order to decide whether or not the solution to LBM is, in fact, an optimal to CIRP, we seek to convert the discrete time solution to LBM into a continuous time feasible solution of the same cost. If successful, then that solution must be an optimal continuous time solution.

Clearly any CIRP feasible solution that performs the same set of vehicle movements as that implied by the LBM solution will do. So a natural approach is to seek a CIRP feasible solution in which that occurs. We divide the process into two steps.

1. **Step 1: extracting delivery routes.** Recall that the LBM solution may not uniquely specify routes for each vehicle, since more than one vehicle may visit a customer at a time. Furthermore, the product quantities assigned to vehicle movements may involve product exchange at a common customer location and time, and so cannot be decomposed to match independent vehicle routes. Thus our first step is to seek a decomposition of the LBM solution into independent vehicle itineraries with associated product flows that provide inventory levels as close as possible to LBM feasibility.

2. **Step 2: revising customer visit times and quantities.** The first step yields a set of vehicle itineraries, each vehicle itinerary specifying a sequence of customer (and depot) visits including the time of the visit and the quantity delivered. The time information implies a sequence of vehicle visits at a customer. In case of multiple visits at the same time, we impose an arbitrary order. Our second step is to revise the time of each visit and the quantity delivered during the visit, while preserving both the vehicle and customer visit sequences and ensuring each vehicle route's timing is feasible. The goal is to ensure that the first customer to run out of product does so as late as possible. If no customer runs out during the planning horizon, then a CIRP solution of the same cost has been found.

We accomplish the first step by solving a MIP and the second step by solving an LP or an IP. Details are found below.

Extracting delivery routes. As discussed earlier, the LBM solution may not uniquely specify routes (or itineraries) for each vehicle. Although this issue may

be overcome with an alternative, vehicle indexed, formulation, we found that this approach did not perform well in practice. Instead, we propose a heuristic for solving the vehicle indexed formulation: first solve the LBM, and then, keeping some aspects of the LBM solution fixed, seek a "nearby" solution to the vehicle indexed formulation. Thus, given a solution to the LBM having vehicle movement variables x^* say, we seek a decomposition of it into independent vehicle itineraries with associated product flows.

To do so, we use a MIP model in which vehicle movement, product flow and delivery quantity variables, x^t_{ijv}, w^t_{ijv} and y^t_{iv}, respectively, are indexed by vehicle, $v \in V := \{1, \ldots, m\}$. The vehicle movement variables are required to decompose those from the LBM solution in the sense that we require $\sum_{v \in V} x^t_{ijv} = x^{*t}_{ij}$ for all $((i,t),(j,t+\tau_{ij})) \in \mathcal{A}^{\mathcal{T}}$. As a consequence, many vehicle movement variables, specifically x^t_{ijv} for which $x^{*t}_{ij} = 0$, can be eliminated in preprocessing, ensuring that the MIP is not difficult to solve in practice.

In other respects, the MIP is very similar to the LBM (adapted to use vehicle indexed product flow and delivery variables). However, since the LBM solution may implicitly require product exchange between vehicles that cannot be decomposed into product flows on independent routes, we cannot guarantee that LBM-feasible inventory levels at customers can be attained. We thus introduce slack and surplus variables, ξ^+_{it} and ξ^-_{it}, respectively, on the inventory level for each customer $i \in N$ and time point $t \in \mathcal{T}$, and seek to minimize a weighted sum of these variables. Although any positive weights would achieve a solution that is in some sense "close to" that of the LBM, we put a higher weight on use of these variables at earlier time points. This is a heuristic designed to maximize the time period over which the resulting routes can feasibly supply customers in the CIRP solution we wish to attain after customer visit times and quantities are revised, subsequently. We call the resulting MIP model

the *vehicle itinerary extraction (VIE)* model:

$$\min \quad \sum_{t\in\mathcal{T}}(T-t)\sum_{i\in N}(\xi_{it}^{+}+\xi_{it}^{-})$$

$$\text{s.t.} \quad \sum_{j\in\delta^{+}(i,t)} x_{ijv}^{t} = \sum_{j\in\delta^{-}(i,t)} x_{jiv}^{t-\tau_{ji}} \qquad (i,t)\in\mathcal{N}^{\mathcal{T}}\setminus\{(0,0),(0,T)\}, v\in V \tag{3.20a}$$

$$\sum_{i\in\delta^{-}(0,T)} x_{i,0,v}^{T-\tau_{i,0}} = 1 \qquad v\in V \tag{3.20b}$$

$$\sum_{v\in V} x_{ijv}^{t} = x_{ij}^{*t} \qquad ((i,t),(j,t+\tau_{ij}))\in\mathcal{A}^{\mathcal{T}} \tag{3.20c}$$

$$\sum_{j\in\delta^{-}(i,t)} w_{jiv}^{t-\tau_{ji}} - \sum_{j\in\delta^{+}(i,t)} w_{ijv}^{t} = y_{iv}^{t} \qquad (i,t)\in\mathcal{N}^{\mathcal{T}}, i\neq 0, v\in V \tag{3.20d}$$

$$0\leq w_{ijv}^{t}\leq Q x_{ijv}^{t} \qquad (i,t)\in\mathcal{N}^{\mathcal{T}}, j\in\delta^{+}(i,t), v\in V \tag{3.20e}$$

$$z_{i}^{t} = z_{i}^{t-1} + \sum_{v\in V} y_{iv}^{t} - u_{i} + \xi_{it}^{+} - \xi_{it}^{-} \qquad i\in N, t\in\mathcal{T}\setminus\{0\} \tag{3.20f}$$

$$z_{i}^{0} = I_{i}^{0} + \sum_{v\in V} y_{iv}^{0} + \xi_{i0}^{+} - \xi_{i0}^{-} \qquad i\in N \tag{3.20g}$$

$$u_{i}\leq z_{i}^{t}\leq C_{i}+u_{i} \qquad \forall i\in N, t=0,\ldots,T-2 \tag{3.20h}$$

$$u_{i}'\leq z_{i}^{T-1}\leq C_{i}+u_{i}' \qquad \forall i\in N, \tag{3.20i}$$

$$y_{iv}^{t}\geq 0 \qquad i\in N, t\in\mathcal{T}, v\in V$$

$$x_{ijv}^{t} \text{ integer} \qquad ((i,t)(j,t+\tau_{ij}))\in\mathcal{A}^{\mathcal{T}}, v\in V$$

$$\xi_{it}^{+},\ \xi_{it}^{-}\geq 0, \qquad i\in N, t\in\mathcal{T},$$

where $u_{i}' = ({}^{H}/_{\Delta} - (T-1))u_{i}$ for each $i\in N$. Note that this formulation does not require depot subtour elimination constraints, since we enforce that each route arrives at the depot exactly once.

Revising Customer Visit Times and Quantities. Consider the routes and customer visits specified by the solution to the VIE model. Let R denote the set of routes and let $\rho(r,k)\in N$ denote the kth customer visited in route $r\in R$,

where n_r (with $n_r \geq 1$) indicates the number of customers visited on route r and $\rho(r,0) = \rho(r,n_r+1) = 0$ (the route starts and ends at the depot). For each customer $i \in N$, let n_i (with $n_i \geq 0$) denote the number of deliveries at customer i. Let $n_0 = 2\sum_{r\in R} n_r$ denote the number of route departures and arrivals, which we will call *events*, at the depot. Let $\phi(r,k)$ for $k \in \{1,\ldots,n_r\}$ denote the visit index in the visit sequence of the kth customer visited in route r. That is, if $\phi(r,k) = \ell$, then the ℓth delivery at customer $\rho(r,k)$ is the kth delivery performed by route r. (If a customer is visited by more than one vehicle at the same time, we arbitrarily order these visits in the customer visit sequence.) For $k = 0$, $\phi(r,0) = \ell$ indicates that the departure of route r from the depot is the ℓth event at the depot. Similarly, for $k = n_r + 1$, $\phi(r,n_r+1) = \ell$ indicates that the arrival of route r at the depot is the ℓth event at the depot. To account for the fact that a vehicles can perform multiple routes in its itinerary, let $r_1^v, r_2^v, \ldots, r_{n_v}^v$ denote the routes performed by vehicle v, where n_v denotes the number of routes performed by vehicle v.

We now construct a linear programming (LP) model to decide (revise) the time of each delivery to a customer, and the quantity to be delivered at that time, while preserving the visit sequence at each customer, the customer sequence on each route and the route sequence in each itinerary. Preserving these sequences, which are encoded in the route indices and the $\rho(\cdot,\cdot)$ and $\phi(\cdot,\cdot)$ functions, enables the timing and quantity decisions to be made using an LP, without the need for binary variables. Naturally, it may be that no feasible CIRP solution using these sequences exists. The LP may be infeasible; solving it is a primal heuristic, and may fail. The LP is constructed so that if it *is* feasible, one of two cases must occur: (1) all its feasible solutions require deliveries by more than one vehicle to a customer at the same time, or (2) a feasible solution in which all vehicle delivery times at a customer are distinct. In the former case, there, again, cannot be a feasible CIRP solution using these sequences. In the latter case, a feasible CIRP solution has been found, and, since the

cost of any solution is purely the sum of the route costs, which are preserved by the model, its solution must be optimal for the CIRP. The LP is constructed as follows.

Let variable t_i^ℓ indicate the time of the ℓth visit to customer i, while q_i^l indicates the quantity delivered in that visit (and $t_i^0 = 0$). Inventory variables, I_i^ℓ, denote the inventory at customer i immediately after the ℓth delivery. (As before, I_i^0 is denotes the initial inventory at i). Let variable t_0^ℓ denote the time of the ℓth event at the depot. Finally, let variable ζ denote the minimum time between deliveries at any customer. We define the LP as follows:

$$
\begin{aligned}
\max \quad & \zeta & & & \\
\text{s.t.} \quad & t^{\phi(r,k)}_{\rho(r,k)} + \hat{\tau}_{\rho(r,k)\rho(r,k+1)} \le t^{\phi(r,k+1)}_{\rho(r,k+1)} & & r \in R,\ k = 0, \ldots, n_r & \text{(3.21a)} \\
& t_0^{\phi(r_j^v, n_{r_j^v}+1)} \le t_0^{\phi(r_{j+1}^v, 0)} & & v \in V,\ j = 1, \ldots, n_v - 1 & \text{(3.21b)} \\
& \sum_{k=1}^{n_r} q^{\phi(r,k)}_{\rho(r,k)} \le Q & & r \in R & \text{(3.21c)} \\
& t_i^\ell \ge t_i^{\ell-1} + \zeta & & i \in N, \ell = 1, \ldots, n_i & \text{(3.21d)} \\
& I_i^\ell = I_i^{\ell-1} - \hat{u}_i \left(t_i^\ell - t_i^{\ell-1} \right) + q_i^\ell & & i \in N,\ \ell = 1, \ldots, n_i & \text{(3.21e)} \\
& I_0^i \ge \hat{u}_i H & & i \in N \text{ and } n_i = 0 & \text{(3.21f)} \\
& I_i^{n_i} - \hat{u}_i (H - t_i^{n_i}) \ge 0 & & i \in N \text{ and } n_i > 0 & \text{(3.21g)} \\
& q_i^\ell \le I_i^\ell \le C_i & & i \in N,\ \ell = 1, \ldots, n_i & \text{(3.21h)} \\
& 0 \le q_i^\ell & & i \in N,\ \ell = 1, \ldots, n_i & \\
& 0 \le t_i^\ell \le H & & i \in N,\ \ell = 1, \ldots, n_i. &
\end{aligned}
$$

Constraints (3.21a) and (3.21b) ensure that for each of the vehicles the visit times at customers properly account for travel times between locations, and, in case a vehicle performs multiple routes, for travel times to and from the depot in between consecutive routes in its itinerary. Constraints (3.21c) ensure that the delivery quantities on a route do not exceed the vehicle capacity. Constraints (3.21d) ensure that

the visit times at customers occur in the same order as in the solution to the lower bound model, and that consecutive deliveries occur at least ζ units of time apart. Constraints (3.21e) ensure that the inventory at customers is accurately modeled. Constraints (3.21f)– (3.21h) together ensure that inventory at each customer is sufficient to meet demand at all points in time. Constraints (3.21f) and (3.21g) ensure that inventory after the last delivery is sufficient to meet demand until the end of the planning horizon, while (3.21h) guarantee that customer i does not run out of product in period $[t_i^{\ell-1}, t_i^{\ell})$ for $\ell = 1, \ldots, n_i$. Constraints (3.21h) also ensure that inventory does not exceed capacity at any customer. If the LP is feasible and has optimal value $\zeta^* > 0$, then the LP solution provides a feasible solution to the CIRP with vehicle movement cost the same as the cost of x^* used in the VIE model. Hence, if x^* is an optimal solution to the LBM, the LP solution also gives an optimal solution to the CIRP.

Theorem 1. *Let the travel times, the storage and vehicle capacities, the initial inventories and the usage rates of an instance of the CIRP be rational. Then, if the CIRP instance is feasible, it has an optimal solution that is rational.*

Proof. Since we have rational parameters, we know that the LP model (3.21) always has a rational optimal solution when it is feasible. The model finds the best visit times and delivery quantities for a given set of vehicle itineraries and customer visit sequences. In particular, if we take the set of vehicle itineraries and customer visit sequences to be those of an optimal solution to the CIRP, then we will get a rational solution. □

Note that this result implies that there exists a discretization of time such that an optimal solution to a time-expanded network formulation using this discretization results in a continuous time optimal solution, which is alluded to in Section 3.4.

A more powerful model is obtained when we only require that the sequence in

which customers are visited in a route is maintained, i.e., when we allow the sequence of routes visiting a customer to change, when we allow the sequence in which routes are performed by a vehicle to change, and when we allow routes to be reassigned to a different vehicle. Doing so, however, requires the introduction of binary variables. Specifically, we let binary variable x_{rk}^{ℓ} indicate whether the k^{th} visit of route r, which is a visit to customer $\rho(r,k)$, is the ℓ^{th} visit at customer $\rho(r,k)$, and let $y_{r\bar{r}}$ indicate that route r and route $\bar{r}$ are performed by the same vehicle and that route $\bar{r}$ is performed immediately after route r by that vehicle.

The visit times have to be viewed from two perspectives: a customer perspective and a route perspective. Let t_i^{ℓ} indicate the time of the ℓ^{th} visit to customer i and let $\bar{t}_r^k$ indicate the time of the k^{th} visit of route r. Similarly, let q_i^{ℓ} indicate the quantity delivered in the ℓ^{th} visit to customer i and $\bar{q}_r^k$ indicate the quantity delivered in the k^{th} visit of route r. We need to ensure that these time are consistent with the decisions we make for the routes. We do so with the following constraints:

$$-H(1-x_{rk}^{\ell}) \leq t_{\rho(r,k)}^{\ell} - \bar{t}_r^k \leq H(1-x_{rk}^{\ell}) \quad r \in R,\ k=1,\ldots,n_r,\ \ell=1,\ldots,n_{\rho(r,k)},$$

$$-Q(1-x_{rk}^{\ell}) \leq q_{\rho(r,k)}^{\ell} - \bar{q}_r^k \leq Q(1-x_{rk}^{\ell}) \quad r \in R,\ k=1,\ldots,n_r,\ \ell=1,\ldots,n_{\rho(r,k)},$$

$$\sum_{\ell=1}^{n_{\rho(r,k)}} x_{rk}^{\ell} = 1 \qquad r \in R,\ k=1,\ldots,n_r,$$

$$\sum_{r\in R}\ \sum_{k=1,\ldots,n_r\,:\,\rho(r,k)=i} x_{rk}^{\ell} = 1 \qquad i \in N,\ \ell=1,\ldots,n_i.$$

Ensuring that each route is assigned to one of the m vehicles, that the routes assigned to a vehicle are performed in sequence, and that a route departures from the depot only after the immediately preceding route has returned to the depot is

enforced by the following constraints

$$\begin{aligned}
\bar{t}_r^{n_r+1} &\leq \bar{t}_{\bar{r}}^0 + H(1 - y_{r\bar{r}}) && r \in R, \bar{r} \in R, r \neq \bar{r}, \\
\sum_{\bar{r} \in R} y_{r\bar{r}} &\leq 1 && r \in R, \\
\sum_{\bar{r} \in R} y_{\bar{r}r} &\leq 1 && r \in R, \\
\sum_{r,\bar{r} \in R} y_{r\bar{r}} &\geq |R| - m, &&
\end{aligned}$$

where the last constraint ensures that at most m routes have no predecessor, which implies that the routes are assigned to no more than m vehicles.

The complete *route-preserving only (RPO)* model can be found in Appendix C. Similar to the previous LP model, if the RPO model is feasible and has an optimal solution with positive value, then, since the cost of any solution is purely the sum of the route costs, which are preserved by both the VIE model and the RPO model, the solution to the latter must be optimal for the CIRP. Computational experiments revealed that even though RPO is an integer program, on instances of interest, it can often find a feasible solution in a short amount of time. Therefore, in our computational study, we have used RPO to try and extract a feasible delivery schedule from a solution to LBM.

3.7 A computational study

The purpose of our computational study is two-fold. First, it is to establish the potential of dynamic discretization discovery algorithms in contexts other than service network design ([8]). Second, and certainly not less important, it is to demonstrate that optimal solutions, or at least provably high-quality solutions, can be obtained for non-trivial instances of one of the most challenging, but practically relevant, variants of the inventory routing problem.

3.7.1 Instances

We define two base instances and then alter those in a controlled way to create additional instances. Based on preliminary experiments, we have seen that the largest instances we can solve in a reasonable amount of time have 15 customers, therefore the base instances have 15 customers. The customers in the two base instances have the same usage rates and storage capacities, but differ in their locations, from which travel times (and costs) are derived.

One of the base instances has customers located in the area $[-5, 5] \times [-5, 5]$ with the company's facility located in the center, so the depot has coordinates $(x_0, y_0) = (0, 0)$. Customer locations are chosen uniformly at random with the given area. The other base instance has customers located in clusters, with locations chosen uniformly at random within three subregions of the area: $[2, 3] \times [2, 3]$, $[2, 3] \times [-2, -3]$, and $[-2, -3] \times [-2, -3]$.

The base usage rate, denoted by u_i^{base} for customer $i \in N$, is a randomly generated integer between 4 and 12 (each equally likely). The customer's storage capacity is an integer that depends on the customer's usage rate: for each $i \in N$, $C_i = f \times u_i$, where f is a randomly generated integer between 8 and 14 (each equally likely). The locations, base usage rates, and storage capacities of the 15 customers are given in Table 3.1. The travel time, $\hat{\tau}_{ij}$, from location i to j, with $i, j = 0, 1, \ldots, 15$, is taken to be the Euclidean distance, $||(x_i, y_i) - (x_j, y_j)||$, rounded to two decimal places, and the cost, c_{ij}, is set equal to the travel time. We note that in all instances, the triangle inequality still holds after rounding the travel times, and that there is sufficient time for a vehicle to reach each customer before it runs out of product.

We generate instances by considering different numbers of customers $n = |N|$. The smallest instance has 5 customers, then 7, 10, 12, and the largest instance has 15 customers. Each instance uses the first n customers in Table 3.1. So for example, a *clustered instance* with 5 customers takes the first 5 customers using the location

coordinates of fourth and fifth columns, while a *non-clustered instance* with 12 customers takes the first 12 customers using the location coordinates of the second and third columns.

Table 3.1: Customer data for the instances.

	Customer locations										
	non-clustered		clustered			Three usage rate cases					
i	x_i	y_i	x_i	y_i	C_i	u_i^{base}	r_i	$1.1u_i^{base}$	r_i	$1.2u_i^{base}$	r_i
1	2.22	-1.53	2.18	2.77	132	11	2	12.1	2	13.2	3
2	3.16	3.67	2.42	-2.46	44	4	2	4.4	2	4.8	2
3	2.68	-2.41	2.98	2.55	64	8	2	8.8	3	9.6	3
4	-1.56	2.12	2.43	-2.75	65	5	1	5.5	2	6.0	2
5	-1.60	0.45	2.59	2.70	120	12	2	13.2	3	14.4	3
6	3.86	3.98	2.40	-2.97	140	10	2	11.0	2	12.0	2
7	-3.45	4.32	2.54	2.03	81	9	2	9.9	2	10.8	3
8	1.23	0.39	2.76	-2.49	44	4	2	4.4	2	4.8	2
9	3.68	-3.73	2.18	2.34	88	11	2	12.1	3	13.2	3
10	-1.86	-1.96	2.49	-2.01	70	7	2	7.7	2	8.4	2
11	3.92	1.97	-2.52	-2.06	100	10	2	11.0	2	12.0	3
12	-1.69	2.82	-2.85	-2.68	112	8	1	8.8	2	9.6	2
13	3.53	1.36	-2.75	-.2.17	132	11	2	12.1	2	13.2	3
14	-2.46	-3.05	-2.48	-2.90	44	4	2	4.4	2	4.8	2
15	-2.66	1.59	-2.74	-2.53	77	6	1	6.6	2	7.2	2

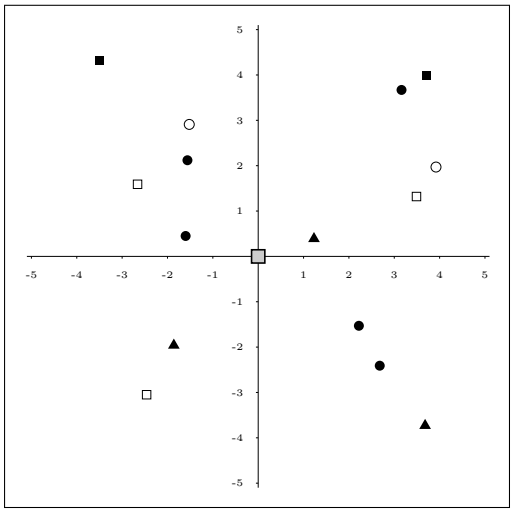

Figure 3.5: Random customers

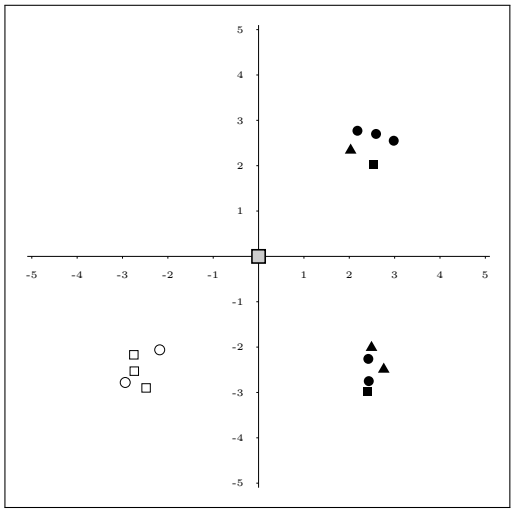

Figure 3.6: Clustered customers

Figures 3.5 and 3.6 plot customers and depot positions. Customers 1 to 5 are

shown with a black circle, Customers 6 and 7 are shown with a triangle (up), Customers 8 to 10 are shown with a square, Customers 11 and 12 are shown with a triangle (down), and, finally, Customers 13 to 15 are shown with a star.

To decide the planning horizon, H, we first observe that the travel time from the depot to any customer and back cannot exceed 15, since $\sqrt{5^2 + 5^2} \approx 7.07$. Thus if the horizon is at least 15, we can be sure there is time to serve each customer once in the planning horizon and return the vehicle to the depot. Also, for simplicity, we set the initial inventory of each customer equal to its storage capacity, so all customers start at full capacity at the start of the planning horizon. From the choice of $C_i = f u_i^{base}$ for $f \leq 14$, we see that provided the horizon is more than 14, each customer will require at least one visit during the planning horizon, using the base data. Since we want a horizon in which some customers will be visited several times, we choose $H = 18$ for all instances. (This is also a choice that enables several alternative integer time discretizations.)

We alter base instances to obtain a larger set of instances by scaling up the usage rates, using the scaling factors 1.1 and 1.2. In other words, we have instances with $\hat{u}_i = u_i^{base}$ for all $i \in N$, $\hat{u}_i = 1.1 u_i^{base}$ for all $i \in N$ and $\hat{u}_i = 1.2 u_i^{base}$ for all $i \in N$, for each of the clustered and non-clustered customer locations and each number of customers. The scaled usage rate cases are shown in the ninth and eleventh columns of Table 3.1. We also observe that the number of visits to customer i over the time horizon must be at least $r_i(\hat{u}_i) = \lceil (\hat{u}_i H - I_i^0) / \min\{C_i, Q\} \rceil$. These lower bounds on the number of visits to a customer, in each of the three usage rate cases, are shown in the eighth, tenth and twelfth columns of Table 3.1, headed r_i. So for the base case, most customers require at least two visits and some require at least one. For the first scaling of usage rates, most customers require at least two visits, while a few require three visits. For the second scaling of usage rates, two or three visits are required, at least, mixed about half and half.

We take the vehicle capacity to be the expected value of the customer capacity, which is the expected value of the base customer usage rate multiplied by the midpoint of the random multiplier range, i.e., $Q = 8 \times 11 = 88$. To investigate the impact of the vehicle capacity, we introduce two variations, one in which we take the vehicle capacity to be $0.75 \times Q = 66$ and one in which we take the vehicle capacity to be $1.25 \times Q = 110$. When we report results, we refer to these three variants as $Q1$ ($Q = 66$), $Q2$ ($Q = 88$), and $Q3$ ($Q = 110$).

Thus, we have $2 \times 3 \times 3 \times 5 = 90$ instances, for the two types of customer location, the three usage rate scalings, the three vehicle capacities, and the five different numbers of customers. Instances will be identified and referenced using a 4-tuple (customer location type, number of customers, usage rate scaling factor, and vehicle capacity scaling factor), e.g., (R,7,U2,Q1), abbreviated as R7U2Q1, indicates an instance with random customer locations, 7 customers, usage rate scaling U2, and vehicle capacity scaling Q1. All instances can be found at `https://github.com/felipelagos/cirplib`.

One of the challenges associated with the LBM is that travel times may be rounded down to zero. In Table 3.2, we show the fraction of travel times in an instance that are rounded to zero for different values Δ (i.e., $\Delta = {}^{H}/_{2k}$ for $k = 1, ..., 9$ and $\Delta = {}^{H}/_{6k}$ for $k = 4, ..., 10$), where, for convenience, we also report the resulting number of time points in the discretization (${}^{H}/_{\Delta}$). Observe that for the random instances and $\Delta \leq {}^{H}/_{30}$, all the travel times are positive, which implies that depot subtour elimination constraints are no longer needed.

Proposition 10 establishes that finding the minimum number of vehicles required to guarantee the existence of a feasible delivery plan is strongly NP-hard. Therefore, to determine the number of vehicles available in an instance, we use a modified version of UBM. Instead of minimizing the total cost of the routes, we minimize the number of vehicles m. We run the UBM with a time limit of 2 hours for each of two values

Table 3.2: Percentage (%) of zero travel times for LBM for a given discretization length.

Length Δ ($^H/_\Delta$)	**Clustered**					**Random**				
	5	7	10	12	15	5	7	10	12	15
9.00 (2)	100.00	100.00	100.00	100.00	100.00	100.00	96.43	96.36	97.44	97.50
4.50 (4)	60.00	60.71	60.00	46.15	40.00	46.67	35.71	43.64	43.59	42.50
3.00 (6)	26.67	32.14	36.36	28.21	25.83	33.33	25.00	25.45	24.36	23.33
2.25 (8)	26.67	32.14	36.36	26.92	25.00	20.00	14.29	12.73	12.82	12.50
1.80 (10)	26.67	32.14	36.36	26.92	25.00	20.00	14.29	10.91	8.97	10.00
1.50 (12)	26.67	32.14	36.36	26.92	25.00	6.67	7.14	5.45	5.13	5.83
1.29 (14)	26.67	32.14	36.36	26.92	25.00	6.67	7.14	3.64	3.85	5.00
1.12 (16)	26.67	32.14	36.36	26.92	25.00	6.67	7.14	3.64	3.85	3.33
1.00 (18)	26.67	32.14	36.36	25.64	24.17	6.67	7.14	3.64	3.85	3.33
0.75 (24)	20.00	25.00	29.09	20.51	19.17	0.00	3.57	1.82	1.28	0.83
0.60 (30)	20.00	17.86	21.82	15.38	14.17	0.00	0.00	0.00	0.00	0.00
0.50 (36)	20.00	14.29	14.55	10.26	10.00	0.00	0.00	0.00	0.00	0.00
0.43 (42)	20.00	14.29	10.91	7.69	6.67	0.00	0.00	0.00	0.00	0.00
0.38 (48)	6.67	7.14	5.45	3.85	4.17	0.00	0.00	0.00	0.00	0.00
0.33 (54)	6.67	7.14	3.64	2.56	2.50	0.00	0.00	0.00	0.00	0.00
0.30 (60)	6.67	7.14	3.64	2.56	1.67	0.00	0.00	0.00	0.00	0.00

of Δ, $^H/_9$ and $^H/_{18}$, recording the best feasible solution found within the time limit. Note that in all cases a feasible solution was found. We then take the minimum of the number of vehicles used in the two solutions. The resulting number of vehicles for each instance can be found in Tables 3.3 and 3.4.

Table 3.3: Minimum number of vehicles for the clustered instance.

		Usage Level								
		U1			U2			U3		
		Capacity			**Capacity**			**Capacity**		
		Q1	Q2	Q3	Q1	Q2	Q3	Q1	Q2	Q3
Customers	5	3	3	3	4	3	3	4	4	3
	7	4	3	3	5	4	4	6	5	4
	10	6	5	4	7	6	5	8	7	6
	12	7	6	5	8	7	6	10	8	7
	15	8	7	6	9	8	7	11	9	8

3.7.2 Experiments

As mentioned above, the purpose of our experiments is to establish the potential of dynamic discretization discovery algorithms and to demonstrate that optimal solutions, or at least provably high-quality solutions, can be obtained for instances of CIRP.

Table 3.4: Minimum number of vehicles for the random instance.

		Usage Level								
		U1			U2			U3		
		Capacity			**Capacity**			**Capacity**		
		Q1	Q2	Q3	Q1	Q2	Q3	Q1	Q2	Q3
Customers	5	3	3	3	4	3	3	4	3	3
	7	5	4	3	6	5	4	6	5	5
	10	7	6	5	8	7	6	9	8	7
	12	8	7	6	9	8	7	11	9	8
	15	9	8	7	11	9	8	13	11	9

In this proof-of-concept study, we simply experiment with different discretizations, i.e., with different values of Δ, and analyze the results. In future research, we will focus on dynamically discovering (location-dependent) discretizations. More specifically, here we solve LBM for $\Delta = {}^{H}/_{k}$ for $k = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 14, 16, 18, 24, 30, 36, 42, 48, 54, 60$, and UBM for $\Delta = {}^{H}/_{k}$ for $k = 9, 10, 12, 14, 16, 18, 24, 30, 36, 42, 48, 54, 60$ (i.e., only finer discretizations). After solving LBM, we use the route-preserving only (RPO) model to try and convert the solution into a CIRP feasible solution. Each model is solved with a time limit of 2 hours.

The lower bound for an instance is the maximum value of best bound over all values of Δ. The upper bound for an instance is the minimum value among all known CIRP feasible solutions (found either when determining the number of vehicles for the instance, or by the UBM, or by the RPO model applied to output of the LBM, for some value of Δ). Tables 3.5 and 3.6 show the resulting optimality gap for each of the instances. When the upper bound is associated with the feasible solution obtained when determining the number of vehicles for the instance, the value of the gap is presented in parentheses. When this happens, neither the UBM nor the RPO model produced a feasible solution.

In Figure 3.7, we summarize these results by means of a histogram that shows the percentage of instances for which a certain optimality gap was achieved. The histogram demonstrates that for most instances, high-quality solutions are obtained,

Table 3.5: Best optimality gap (%) for clustered instances.

		Usage Level								
		U1			U2			U3		
		Capacity			**Capacity**			**Capacity**		
		Q1	Q2	Q3	Q1	Q2	Q3	Q1	Q2	Q3
Customers	5	0.00	0.00	0.00	0.00	1.16	0.00	1.94	0.00	0.00
	7	1.12	4.43	0.94	0.00	1.75	1.44	0.00	0.00	1.82
	10	0.00	2.14	2.47	0.71	2.09	3.29	0.84	1.25	2.39
	12	0.00	0.87	2.24	3.13	1.78	2.15	0.51	8.54	6.82
	15	0.00	3.04	7.02	(13.59)	3.21	5.94	3.65	7.74	(24.17)

Table 3.6: Best optimality gap (%) for random instances.

		Usage Level								
		U1			U2			U3		
		Capacity			**Capacity**			**Capacity**		
		Q1	Q2	Q3	Q1	Q2	Q3	Q1	Q2	Q3
Customers	5	0.00	0.00	8.63	0.00	0.25	0.29	0.00	2.55	0.00
	7	0.00	0.00	8.80	0.00	0.00	0.66	0.00	0.04	3.57
	10	4.15	0.00	5.17	0.00	3.77	0.00	2.32	3.05	6.13
	12	8.54	0.01	5.26	(11.09)	7.46	9.49	0.01	14.38	(24.16)
	15	(18.36)	1.47	(17.62)	(15.62)	(20.25)	(34.39)	7.06	(26.04)	(18.19)

especially for clustered instances; one clustered instance with 15 customers was solved to optimality. Twelve out of 90 instances have an optimality gap of more than 10%; most of them random instances with 15 customers.

It is not surprising that when neither the UBM nor the RPO model finds a feasible solution, the resulting gap is large (when we determine the number of vehicles for an instance, we minimize the number of vehicles and do not consider costs). To assess the impact of the available number of vehicles on the ability to obtain low cost solutions, we solved the largest instances with random customer locations assuming one more vehicle was available. The results can be found in Table 3.7. We see that for all instances a feasible solution was obtained, and that except for Instance R15U2Q2 these solutions are either optimal or close to optimal.

In Figure 3.8, we focus on the impact of the discretization (i.e., the value of Δ/H) on the bounds for a few select instances. For each value of Δ/H, we show the value of the bound on the cost obtained by solving LBM ("Lower Bound" in the legend), the cost of the feasible delivery schedule extracted by RPO from the solution to LBM,

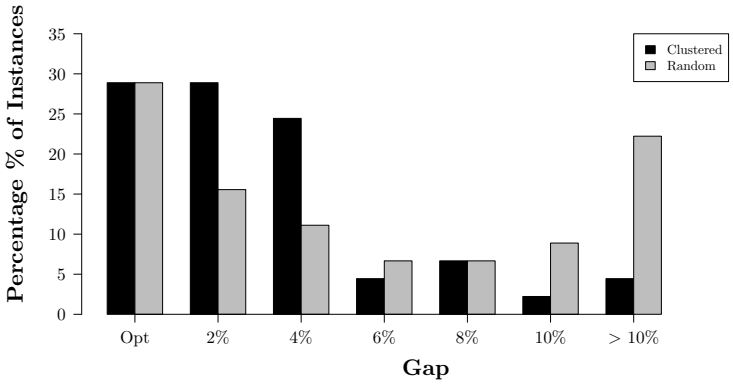

Figure 3.7: Histogram of instances achieving a certain optimality gap.

Table 3.7: Best optimality gap (%) for random instances with one more vehicle than the best known minimum.

		Usage Level								
		U1			U2			U3		
		Capacity			**Capacity**			**Capacity**		
		Q1	Q2	Q3	Q1	Q2	Q3	Q1	Q2	Q3
Customers	15	2.05	0.01	0.12	1.62	14.12	0.98	0.72	2.18	9.45

if any ("LBM Feas" in the legend), and the cost of the feasible delivery schedule obtained when solving UBM, if any ("UBM Feas" in the legend). In Appendix D, we provide, for four of these instances, detailed solution statistics (column headings are self-explanatory) for all values of H/Δ.

The most striking observation is that for all these instances, the best lower bound is obtained for the largest value of Δ, i.e., for the discretization with the fewest number of time points. This is both disappointing, because one would expect that a finer discretization should lead to a better bound, but also encouraging, because it suggests that with carefully chosen time points, it may be possible to get good bounds also for larger instances. We also observe that if successful, the RPO model produces high-quality solutions, often optimal solutions, and that the UBM produces many feasible solution, but that their quality is not always high.

To highlight the fact that these instances are non-trivial, we investigate the opti-

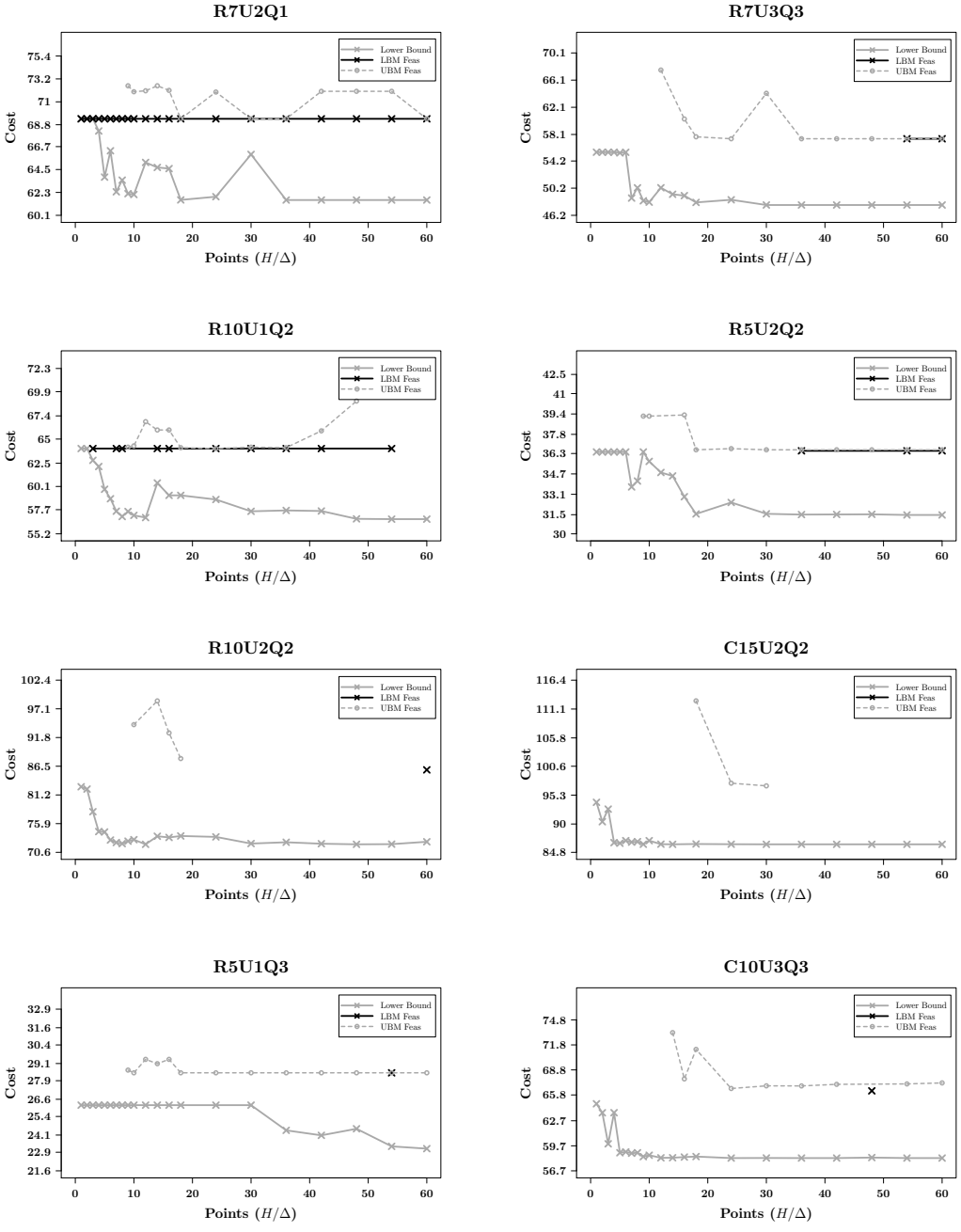

Figure 3.8: Lower and upper bound value for different values of $\frac{H}{\Delta}$.

mal solution for Instance R7U2Q1. The vehicle itineraries can be found in Table 3.8 and the customer inventory profiles can be found in Figure 3.9. We see that three

Table 3.8: Vehicle itineraries in the optimal solution to Instance R7U2Q1.

Vehicle	Route			Visit 1			Visit 2			Visit 3		
		t_0	t_{n+1}	i	t_i	q_i	i	t_i	q_i	i	t_i	q_i
1	1	0.00	8.15	1	5.45	66.00						
	2	8.15	18.00	3	11.75	46.20	1	12.74	19.80			
2	1	0.00	18.00	4	4.06	22.34	7	7.00	31.20	4	14.59	11.66
3	1	0.00	5.57	5	3.91	51.60						
	2	5.57	18.00	6	11.01	58.00						
4	1	0.00	18.00	2	8.72	35.20						
5	1	0.00	9.79	3	6.19	54.46						
	2	9.79	18.00	5	13.00	10.24	5	13.78	55.76			
6	1	0.00	18.00	7	9.82	66.00						

vehicles make multiple trips during the planning horizon and that one vehicle delivers product at four customers on a single trip. Furthermore, we see that all customers receive multiple deliveries during the planning horizon and that one customer receives two consecutive deliveries from a vehicle that waits at its premises. We note that an alternative optimal solution exists that combines the two deliveries into a single delivery (at the time of the first visit). This highlights one of the challenges in solving CIRP instances. There may be many alternative solutions with the same cost.

We presented (and used) a number of valid inequalities to strengthen the linear programming relaxations of both the LBM and the UBM. To show the importance of using these valid inequalities, in Figure 3.10, we show for Instance R7U2Q1 the best lower and upper bound value for different values of $^{H}/_{\Delta}$ when solving the LBM and the UBM with and without the valid inequalities (a $^{+}$ in the legend is used to indicate that the results are obtained when valid inequalities are added to the formulation).

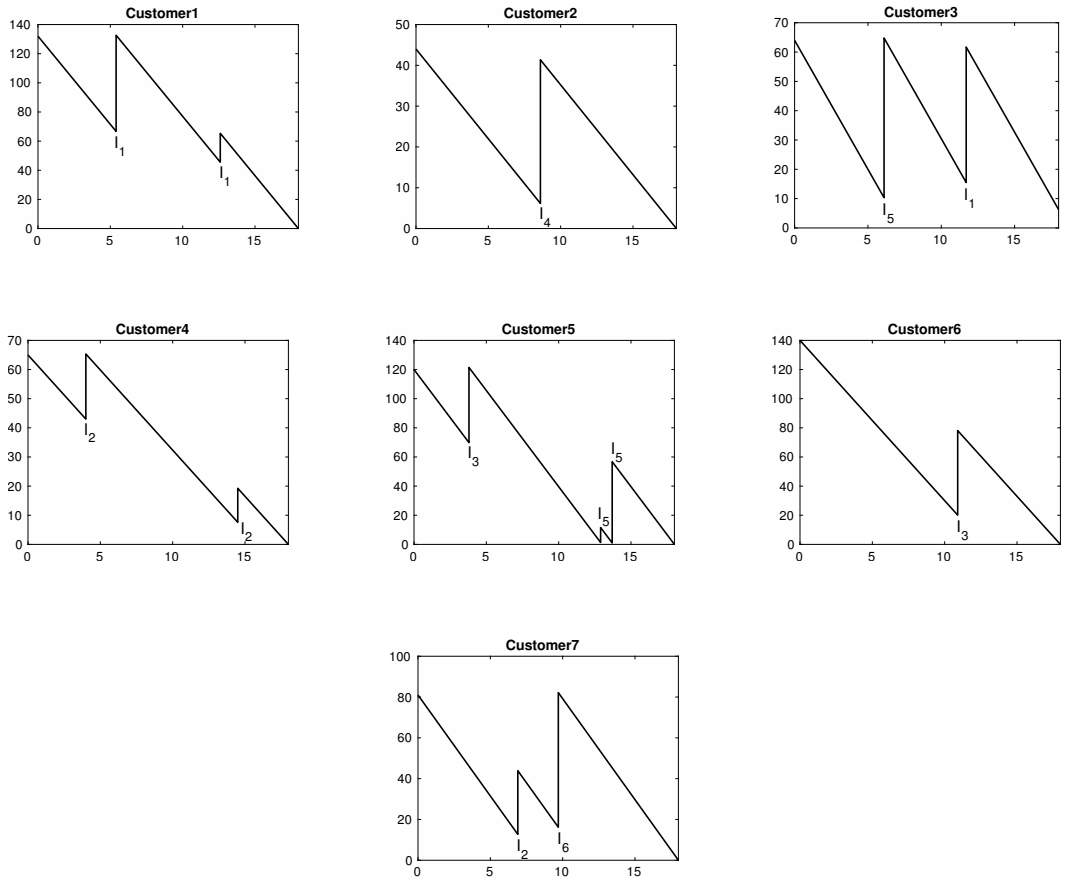

Figure 3.9: Customer inventory profiles in the optimal solution to Instance R7U2Q1.

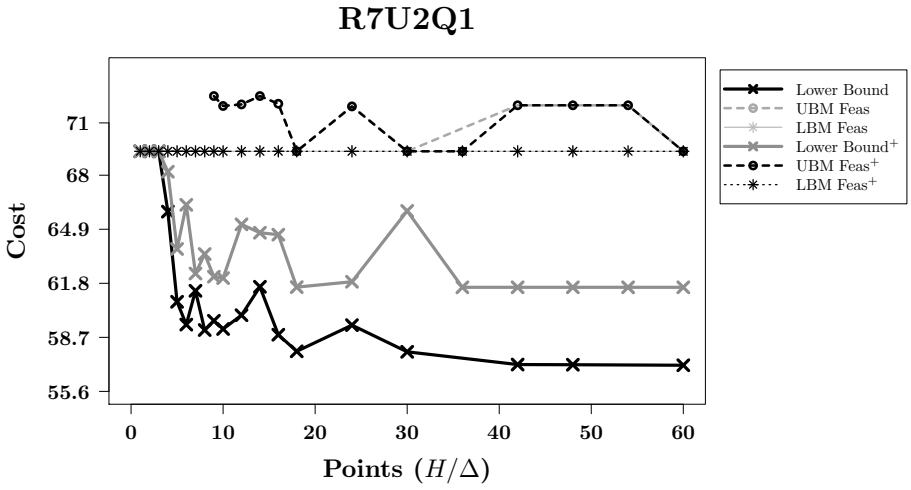

Figure 3.10: Assessing the value of the valid inequalities for Instance R7U2Q1.

We observe that incorporating the valid inequalities results in improved lower and upper bounds for many values of H/Δ. Furthermore, although not visible in the figure, two more feasible delivery schedules were extracted by RPO when starting from solutions to LBM with valid inequalities. (We note that for this particular instance, an optimal solution was found even without using any valid inequalities.)

3.8 Discussion and future research

In this chapter, we have demonstrated that proven optimal solutions to instances of the continuous time inventory routing problem can be obtained using relatively simple time discretization ideas in combination with sophisticated integer programming models. This achievement relies on an integer program that provides a lower bound on the optimal solution value, an integer program that extracts a set of delivery routes from a solution to the lower bound model, and an integer program that seeks to manipulate a set of extracted delivery routes so as to construct a feasible continuous-time solution.

The computational study has revealed that the integer program that provides a

lower bound quickly becomes very difficult to solve as the number of time points in the discretization increases. The best lower bounds are typically found for very coarse time discretizations; for finer time discretizatons, the integer program does not solve to optimality within the given time limit and only the best bound can be used. Thus, at the moment, the best strategy for obtaining a lower bound for an instance of the CIRP is to solve the integer program that provides a lower bound for a few values of $\Delta = {}^{H}/_{k}$ with $k \leq 4$.

This, of course, is not entirely satisfactory, which is why we are currently pursuing a full-fledged dynamic discretization discovery algorithm. The solution of the integer program that provides a lower bound for a specific of Δ, can be viewed as a single iteration of a dynamic discretization discovery algorithm. What is missing is a component that analyzes why a set of extracted delivery routes cannot be converted to a feasible (and therefore optimal) continuous-time solution, and uses the results of that analysis to refine the time discretization.

To ensure that a computationally efficient dynamic discretization discovery algorithm results, it is necessary that non-uniform time discretizations can be handled, i.e., the set of time points associated with a location in the partially time-expanded network can depend on the location and the time between two consecutive time points associated with a location can vary. Fortunately, it is not too difficult to extend the lower and upper bound models presented in Sections 3.5.2 and 3.6.1 to handle non-uniform discretizations.

Thus, to develop a full-fledged dynamic discretization discovery algorithm that can handle larger CIRP instances, what remains is the design and implementation of a component that analyzes why a set of extracted delivery routes cannot be converted to a feasible continuous-time solution, and uses the results of that analysis to identify time points that can be added to the set of time points at one or more locations and that ensure that the solution to the integer program that produces a lower bound

improves. This is easier said than done, and is the focus of our current research.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. 1662848.

CHAPTER 4
AN EXACT ALGORITHM FOR THE CONTINUOUS TIME INVENTORY ROUTING PROBLEM WITH OUT-AND-BACK ROUTES

4.1 Introduction

The Inventory Routing Problem (IRP) integrates inventory management, vehicle routing, and delivery scheduling decisions. The IRP arises in the context of Vendor Managed Inventory (VMI), in which a supplier makes the replenishment decisions for products delivered to its customers. The variant of interest in this chapter was introduced in the seminal chapter by [2]. Critical characteristics of this variant are that only transportation costs are considered and that the system evolves in continuous time. That is, the amount of product that can be delivered to a customer at a particular point in time depends on the storage capacity and inventory at that point in time (which depends on the initial inventory, the product usage rate, and the time elapsed since the start of the planning period, and on the amount of product delivered since the start of the planning period). As a consequence, delivery times have to be scheduled carefully and vehicle travel times have to be accounted for accurately. This contrasts with the majority of the variants of the IRP considered in the literature, where the planning horizon is partitioned into periods and where it is assumed that delivery routes take place at the start of the period, product consumption takes places at the end of the period, and that both happen instantaneously. For more comprehensive introductions to and discussions of the IRP, see [62] and [5].

The variant considered in this chapter is motivated by the IRP encountered by companies in the liquid gas industry, e.g., Air Liquide (`www.airliquide.com`), Air

Products (www.airproducts.com), and Praxair (www.praxair.com). These companies produce liquefied gases, e.g., liquid oxygen, liquid nitrogen, or liquid argon, install tanks on their customers' premises, and guarantee minimum product availability at any time. Customers use (consume) product at a certain rate (which can differ at different times of the day) often 24 hours per day (e.g., liquid oxygen in hospitals.) Thus, the amount of product that can be delivered to the tank changes at the same rate. The companies continuously monitor product usage and tank inventory levels so that they can produce cost-effective delivery schedules that meet their service commitments (i.e., the guaranteed minimum product availabilities). In practice, the companies tend to have customers that require multiple deliveries per day as well as customers that require as few as one or two deliveries per week. As a consequence, the use of a continuous time variant of the IRP is most appropriate in these settings because it provides the most accurate representation of the system. Also relevant is the fact that the company contracts typically specify that the customers own/purchase the product upon delivery, which means that the companies do not have to consider product holding costs at the customer's location. This variant of the IRP has attracted attention in the past, e.g., [4], [63], and [64, 65], and, was recently considered interesting and challenging enough to form the ROADEF/EURO 2016 Challenge (for more information, see www.roadef.org/challenge/2016/en/sujet.php) with real-life data provided by Air Liquide.

Solving instances of any variant of the IRP is challenging [66]. Integer programming techniques have been used for the "period" variants, e.g., branch-and-cut [67, 68] and branch-and-cut-and-price [67, 69]. The most advanced and successful of these can now solve instances with up to 50 customers and up to 5 vehicles. For the "continuous time" variant, lower bounding techniques for this problem have been developed in [70]. The Continuous Time IRP (CIRP) was formally presented in [72], and complex integer and linear models are studied to find provable optimal solutions for the

problem.

The CIRP has been shown to be a difficult problem. In [72], instances with up to 15 customers can be solved, but finding an optimal solution for the problem is not always possible. Given the complexity of the ideas we study, we consider a simpler setting for the continuous time IRP: the CIRP with out-and-back routes only (CIRP-OB). In this problem, the vehicles depart from the depot, visit only one customer, and then return. Only direct deliveries are allowed. For the IRP, it is natural that this direct strategy is considered first because it is easy-to-implement and is frequently used in industrial distribution systems. The IRP with direct deliveries has drawn attention in the IRP literature [73, 74]. Studies show direct deliveries are effective compared to any other feasible distribution strategy in the long-run [75, 76, 77]. In [77], the authors derived an explicit formula for evaluating how effective the direct delivery strategy is regarding the long-run average cost, concluding that it is at least 94% effective whenever the customer capacities exceed 71% of the vehicle capacity. In [76], the effectiveness of the direct deliveries can be represented by a function of some system parameters. Under some conditions on the usage rate and vehicle capacity, direct deliveries is an optimal distribution strategy.

As we find in this study, modeling inventory in continuous time and accurately modeling vehicle travel times make for a particularly challenging IRP. Indeed, optimization problems over continuous time, in general, have been found to be difficult to solve to optimality. Compact models that use continuous variables to model time have weak linear programming (LP) relaxations. Their solution with current integer programming solver technology is limited to only small instances. Extended formulations, with binary variables indexed by time, have much stronger relaxations, but (tend to) have a huge number of variables. Such formulations rely on a *discretization* of time, which introduces approximation. Recently, [8] introduced a Dynamic Discretization Discovery (DDD) algorithm for solving the continuous time service net-

work design problem, which uses extended integer programming formulations. The key to the approach is that it discovers exactly which times are needed to obtain an optimal, continuous-time solution, in an efficient way, by solving a sequence of (small) integer programs. The integer programs are constructed as a function of a subset of times, with variables indexed by times in the subset. These IPs are carefully designed to be tractable in practice and to yield a lower bound on the optimal continuous-time value. Once the *right* (very small) subset of times is discovered, the resulting integer programming model yields the continuous-time optimal value.

Our contributions in this chapter are both theoretical and algorithmic. We

- successfully develop and implement the DDD algorithm; adapt a mixed integer program to yield lower bounds on the optimal CIRP-OB value, develop a mixed integer program to repair solutions to the lower bound model, and put forward routines to improve the time-expanded network when the lower bound solution cannot be converted;
- show that the DDD algorithm is an exact algorithm for the CIRP-OB, proving that for a feasible CIRP-OB instance, the algorithm finishes with an optimal solution;
- derive conditions on the waiting times and arrival visit times that a subset of CIPR-OB optimal solutions hold (so we restrict our search to that subset);
- show how to incorporate the previous conditions into the lower bound formulation (strengthening the model); and,
- generate a new set of instances for the CIRP-OB to carry out a computational study, and show that our DDD algorithm is able to produce provable optimal solutions for instances with up to 30 customers.

The remainder of the chapter is organized as follows. In Section 4.2, we formally introduce the continuous time inventory routing problem with out-and-back routes only, and we show that this problem is *strongly NP-Hard*. In Section 4.3, we provide a mixed integer programming formulation for the CIRP-OB over a time discretization. In Section 4.4, we present some conditions that a subset of CIRP-OB optimal solutions hold. In Section 4.5, we show how the mixed integer programming formulation can be modified to produce a lower bound on the cost of an optimal delivery plan and we present valid inequalities that strength the formulation. In Section 4.6, we outline two approaches for constructing feasible delivery plans. In Section 4.7, we describe the DDD algorithm and show that it is an exact algorithm for the CIRP-OB. In Section 4.8, we present and discuss the results of an extensive computational study. Finally, in Section 4.9, we discuss relevant aspects of the algorithm and models presented and describe the next steps towards developing a DDD algorithm for the CIRP, i.e., in which routes can visit more than one customer.

4.2 The Continuous Time Inventory Routing Problem with Out-and-Back Routes

We consider a vendor managed resupply environment in which a company manages the inventory of its customers, resupplying them from a single facility.

Each customer i in the set $N = \{1, \ldots, n\}$ of customers has local storage capacity C_i, uses product at a constant rate u_i, and has initial inventory I_i^0 at the start of the planning period, which is assumed to be equal or less than C_i. The planning horizon is H. The company deploys a fleet of m homogeneous vehicles, each with capacity Q, to deliver product to its customers. We consider an out-and-back routes only setting: a vehicle route starts at the depot visits a single customer and returns to the depot. Travel times τ_i and travel costs c_i, between the depot and a location $i \in N$, are assumed to be symmetric. All parameters are strictly positive. The set of

customers and the depot corresponds to $N_0 = N \cup \{0\}$.

We allow a vehicle to wait at a customer location and to make multiple deliveries while at the customer's premises. This may be beneficial as it allows delivery of more than the customer's storage capacity without the need for an intermediate trip to the depot. In practice, a customer may have sufficient space for several vehicles to wait at their premises, but usually at most one vehicle can deliver product at a time. Although the time needed for a vehicle to deliver product may depend on the quantity to be delivered, there is usually a substantial overhead time needed to engage and disengage the delivery equipment. Thus the delivery time can be reasonably well approximated by a constant (possibly customer-dependent) length of time. This situation can be modeled by the use of two locations for each customer, one for parking and one for making a delivery at the customer, with the latter constrained to allow at most one vehicle at a time. However, given the complexity of the ideas we wish to discuss in this chapter, we make the simplifying assumption that all locations have the single-vehicle constraint: we assume that it is *not* possible for multiple vehicles to visit the same customer at the same time. We further assume that product delivery at a customer site is instantaneous. It is not difficult to extend the ideas we present here to account for multiple vehicles waiting at a customer and constant delivery time. We also assume that there is no cost for waiting.

Vehicles are assumed to be at the company's facility at the start of the planning period and have to be back at the company's facility at the end of the planning period. However, vehicles can make multiple trips during the planning period. We assume that the loading of a vehicle at the company facility is instantaneous.

We assume that the company does not incur any holding cost for product, either at the company facility or at any of the customer sites. We also assume that the company facility always has sufficient product to supply customers; it does not have either production or storage capacity constraints.

The goal is to find a minimum cost delivery plan that ensures that none of the customers runs out of product during the planning period. A delivery plan specifies a set of vehicle itineraries, each of which consists of a sequence of out-and-back routes that start and end at the company facility within the time horizon, to be performed by a single vehicle. Each route specifies a departure time from the facility, a visit to a customer, the delivery times and quantities delivered to the customer at those times and a departure time from the customer location. We refer to this problem as the Continuous Time Inventory Routing Problem with Out-and-Back routes (CIRP-OB). Next, even though we assume only out-and-back routes, we show that the problem is still *strongly NP-Hard*.

Proposition 16. *The CIRP-OB is* strongly NP-Hard.

Proof. Reduction for 3-PARTITION. Given an instance of 3-PARTITION, i.e., a set of items $\{a_1, a_2, \ldots, a_{3n}\}$ with $\sum_{i=1}^{3n} a_i = nB$ and $\frac{1}{4}B < a_i < \frac{1}{2}B$ for $i = 1, \ldots, 3n$, construct the following instance of CIRP-OB. The instance has $3n$ customers. Customer i is located at travel time $\tau_i = \frac{1}{2}a_i$ from the depot, has usage rate $u_i = 1$, storage capacity $C_i = B$, and initial inventory $I_i^0 = B$âĹŠϵ. There are n vehicles with capacity $Q = 3B$. The planning horizon is $H = B$. Each customer needs at least one delivery during the planning horizon, because the usage during the planning horizon is $B >$ BâĹŠϵ. Each vehicle can make at most 3 deliveries during the planning horizon, because the time required to make 4 deliveries is strictly greater than $4 \times \frac{1}{4}B = B = H$, which implies that the maximum number of deliveries that can be made during the planning horizon is $3n$ (there are n vehicles). This implies that each vehicle has to make 3 deliveries. Because $\sum_{i=1}^{3n} a_i = nB$, this implies that each vehicle k must have $\sum_{i \in S_k} a_i = B$, where S_k is the set of customers visited by vehicle k. Thus, a feasible delivery schedule exists if and only if the instance of 3-PARTITION is a *yes*-instance. □

4.3 Exact Time Indexed Formulation

In [72] the authors prove that a rational optimal solution always exists for a feasible Continuous Time IRP (CIRP) instance. The CIRP-OB is a special case of the CIRP, thus an optimal solution whose deliveries and decision times are rational is guaranteed as long as a feasible solution exists. Then, there exists a time discretization sufficiently fine in which a time indexed formulation finds an optimal solution. In this section we assume a time discretization with the property just mentioned is given and we present a time indexed formulation on it.

Let $K_i + 1$ be the number of time points in the time discretization of location $i \in N_0$, and let $\mathcal{T}_i = \{0, ..., K_i\}$ be the set of indexes. Let $\{t_i^k\}_{k \in \mathcal{T}_i}$ be the set of time points and we assume the times are ordered, $t_i^{k-1} < t_i^k$, $k \geq 1$. We also assume that $\{0, H\} \subseteq \{t_i^k\}_{k \in \mathcal{T}_i}$. Note that these time point sets can represent any time discretization, i.e., the difference between two consecutive times, $t_i^{k+1} - t_i^k$, can be any rational number, $i \in N_0$, $k \in \mathcal{T}_i$. We consider a time-expanded network that preserves travel times, and so, the following conditions to all time points:

- for all $i \in N$ and $k \in \mathcal{T}_0$ such that $t_0^k + \tau_i \leq H$, then $t_0^k + \tau_i \in \{t_i^\ell\}_{\ell \in \mathcal{T}_i}$;
- for all $i \in N$ and $k \in \mathcal{T}_0$ such that $t_0^k - \tau_i \geq 0$, then $t_0^k - \tau_i \in \{t_i^\ell\}_{\ell \in \mathcal{T}_i}$;
- for all $i \in N$ and $k \in \mathcal{T}_i$ such that $t_i^k + \tau_i \leq H$, then $t_i^k + \tau_i \in \{t_0^\ell\}_{\ell \in \mathcal{T}_0}$;
- for all $i \in N$ and $k \in \mathcal{T}_i$ such that $t_i^k - \tau_i \geq 0$, then $t_i^k - \tau_i \in \{t_0^\ell\}_{\ell \in \mathcal{T}_0}$.

Locations in N_0 and times $\{t_i^k\}_{i \in N_0, k \in \mathcal{T}_i}$ induce a time-expanded network. This network consists of the nodes set $\mathcal{N}^{\mathcal{T}} = \{(i, t_i^k) : i \in N_0, k \in \mathcal{T}_i\}$ and the timed arcs

set,

$$\mathcal{A}^{\mathcal{T}} = \left\{ a_{ij}^{k\ell} := \Big((i,t_i^k),(j,t_j^\ell)\Big) \in \mathcal{N}^{\mathcal{T}} \times \mathcal{N}^{\mathcal{T}} : \begin{array}{r} \text{if } i = 0 \text{ and } j \in N \text{ then } t_0^k + \tau_j = t_j^\ell \\ \text{if } i \in N \text{ and } j = 0 \text{ then } t_i^k + \tau_i = t_0^\ell \\ \text{if } i = j \text{ and } k < K_i \text{ then } t_j^\ell = t_i^{k+1} \end{array} \right\}.$$

Let $\delta^+(i,t_i^k) = \{(j,\ell) : \exists \ell, a_{ij}^{k\ell} \in \mathcal{A}^{\mathcal{T}}\}$ be the set of locations and time indexes that can be reached from i at time t_i^k. Similarly, we define $\delta^-(i,t_i^k) = \{(j,\ell) : \exists \ell, a_{ji}^{\ell k} \in \mathcal{A}^{\mathcal{T}}\}$, as the set of locations and time indexes that reach the node $(i,t_i^k) \in \mathcal{N}^{\mathcal{T}}$. To simplify notation, we define $\bar{u}_i^k = u_i(t_i^k - t_i^{k-1})$ as the product consumed by a customer $i \in N$ during the time interval $[t_i^{k-1}, t_i^k]$, for $(i,t_i^k) \in \mathcal{N}^{\mathcal{T}}$ and $k \geq 1$.

We consider the binary variables x_{ij}^k, representing whether a vehicle travels from i to j at time t_i^k, $x_{ij}^k = 1$, or not, $x_{ij}^k = 0$; for all arc $a_{ij}^{k\ell} \in \mathcal{A}^{\mathcal{T}}$. The variable w_{ij}^k indicates the amount of product that flows for location i to j at time t_i^k. Let the variable y_i^k be the delivered product and z_i^k be the inventory level at customer $i \in N$ at time t_i^k, $(i,t_i^k) \in \mathcal{N}^{\mathcal{T}}$. We assume that z_i^k is the inventory customer level after the delivery, if any, takes place.

$$\min \quad \sum_{i \in N} \sum_{k \in \mathcal{T}_0} 2c_i x_{0i}^k,$$

$$\text{s.t.} \quad x_{i0}^k + x_{ii}^k = x_{ii}^{k-1} + x_{0i}^\ell, \qquad i \in N, k \in \mathcal{T}_i, (0,\ell) \in \delta^-(i, t_i^k), \quad (4.1a)$$

$$x_{00}^k + \sum_{i \in N} x_{0i}^k = x_{00}^{k-1} + \sum_{(i,\ell) \in \delta^-(0,t_0^k)} x_{i0}^\ell, \quad k \in \mathcal{T}_0 \setminus \{0, K_0\}, \quad (4.1b)$$

$$x_{00}^0 + \sum_{i \in N} x_{0i}^0 = m, \quad (4.1c)$$

$$w_{0i}^\ell + w_{ii}^{k-1} - w_{ii}^k = y_i^k, \qquad i \in N, k \in \mathcal{T}_i, (0,\ell) \in \delta^-(i, t_i^k), \quad (4.1d)$$

$$x_{ii}^k + x_{i0}^k \leq 1, \qquad i \in N, k \in \mathcal{T}_i, \quad (4.1e)$$

$$0 \leq w_{ij}^k \leq Q x_{ij}^k, \qquad a_{ij}^{k\ell} \in \mathcal{A}^\mathcal{T}, \quad (4.1f)$$

$$z_i^k = z_i^{k-1} + y_i^k - \bar{u}_i^k, \qquad i \in N, k \in \mathcal{T}_i \setminus \{0\}, \quad (4.1g)$$

$$z_i^0 = I_i^0 + y_i^0, \qquad i \in N, \quad (4.1h)$$

$$\bar{u}_i^{k+1} \leq z_i^k \leq C_i, \qquad i \in N, k \in \mathcal{T}_i, \quad (4.1i)$$

$$0 \leq y_i^k, \qquad i \in N, k \in \mathcal{T}_i,$$

$$x_{ij}^k \in \{0,1\}, \qquad a_{ij}^{k\ell} \in \mathcal{A}^\mathcal{T}.$$

Constraints (4.1a), (4.1b) and (4.1c) ensure vehicle flow balance and enforce that all m vehicles are returned to the depot at the end of the planning horizon. Constraints (4.1d) impose product flow balance and ensure that the product arriving at a customer is either delivered at that customer or remains on the vehicle. Note that no product can come back to the depot. Constraints (4.1e) together with the requirement that each x_{ij}^t variable is binary ensure that at most one vehicle can be visiting a customer at any one time. Constraints (4.1f) link the product flows to the vehicle flows. Constraints (4.1g) and (4.1h) model product usage at a customer and inventory balance. Constraints (4.1i) ensure that inventory at a customer is sufficient, after each delivery, to meet the customer demand in the coming period and never exceeds

the local storage capacity.

A lower bound value to the linear relaxation of formulation (4.1) can be found using the parameters of the problem. This value corresponds to the cost of the minimum number of vehicles required (fractional) to deliver the product customers need to cover their usage during the planning horizon. In proposition 18, next in this section, we show that this lower bound is tight to the linear relaxation when the number of vehicles is large: there exists a feasible solution whose cost is (4.2).

Proposition 17. *A lower bound for the formulation* (4.1) *linear relaxation optimal value is given by*

$$\sum_{i \in N} 2c_i \left(\frac{Hu_i - I_i^0}{Q} \right). \tag{4.2}$$

Proof. Note that the inventory balance constraints imply the following condition on the total delivery for any $i \in N$,

$$z_i^{K_i} = I_i^0 + \sum_{k \in \mathcal{T}_i} y_i^k - Hu_i \geq 0,$$

$$\sum_{k \in \mathcal{T}_i} y_i^k \geq Hu_i - I_i^0.$$

Also, note that summing up the product balance constraints (4.1d) leads to,

$$\sum_{\ell \in \mathcal{T}_0} w_{0i}^\ell = \sum_{k \in \mathcal{T}_i} y_i^k,$$

and since $w_{ij}^k \leq Q x_{ij}^k$, we have,

$$Hu_i - I_i^0 \leq \sum_{k \in \mathcal{T}_i} y_i^k = \sum_{\ell \in \mathcal{T}_0} w_{0i}^\ell \leq Q \sum_{\ell \in \mathcal{T}_0} x_{0i}^\ell.$$

Multiplying by $2c_i$ and summing up over all $i \in N$, we get that any feasible solution to formulation (4.1) is greater than or equal to (4.2). □

Proposition 18. *Consider a feasible CIRP-OB instance and* $m = \sum_{i \in N} \lceil (C_i - I_i^0 + 2\tau_i u_i)/Q \rceil$. *Also, assume that the time points* $H - C_i/u_i$ *and* τ_i *are in* $\{t_i^k\}_k$, *for all* $i \in N$. *The optimal solution value to formulation* (4.1) *linear relaxation is equal to* (4.2), $\sum_{i \in N} 2c_i \left(\frac{Hu_i - I_i^0}{Q} \right)$.

Proof. We show a feasible solution whose cost is equal to (4.2).

For all $i \in N$, consider $k = k_1, k_1 + 1 \ldots, k_2$, with $t_i^{k_1} = \tau_i$ and $t_i^{k_2} = H - C_i/u_i$. Also, let $\bar{k} = \text{argmin}_{k=k_1,k_1+1\ldots,k_2}\{I_i^0 + Q(k + 1 - k_1) - u_i t_i^k > C_i\}$. We suggest the following solution:

$$x_{0i}^{\ell} = \begin{cases} 1, & k = k_1, \ldots, \bar{k} - 1, \\ \frac{C_i - (I_i^0 + Q(\bar{k} - k_1) - u_i t_i^{\bar{k}})}{Q}, & k = \bar{k}, \\ \frac{\bar{u}_i^k}{Q}, & k = \bar{k} + 1, \ldots, k_2, \end{cases} \quad \text{with } \ell \in \delta^-(i, t_i^k).$$

In this solution, each customer $i \in N$ receives full vehicle deliveries at each time t_i^k, $k = k_1, \ldots, \bar{k}$, until the inventory is equal to C_i and then, after time $t_i^{\bar{k}+1}$, the deliveries are equal to what is consumed in a time interval $\bar{u}_i^k$. There is no waiting, thus all variables x_{ii}^k are equal to zero. Constraints (4.1a)-(4.1b) are satisfied because we have constructed the x variables as a flow. The number of vehicles constraint (4.1c) is guaranteed by the number m we've assumed; the maximum number of vehicles a customer needs in this solution is at most $\lceil (C_i - I_i^0 + 2\tau_i u_i)/Q \rceil$, corresponding to the product usage while a vehicle is traveling and the initial product needed to fill the inventory level, $C_i - I_i^0$.

Note that in this solution the inventory levels are feasible for all $i \in N$ and $k = k_1, \ldots, k_2$,

$$z_i^k = \begin{cases} z_i^{k-1} + Q - \bar{u}_i^k, & k = k_1, \ldots, \bar{k} - 1, \\ C_i, & k = \bar{k}, \ldots, k_2, \end{cases}$$

and since it also holds that $t_i^{k_2} = H - C_i^0/u_i$, then inventory levels are feasible for all planning horizon.

We conclude this solution is feasible and its total cost is exactly (4.2),

$$\begin{aligned}\sum_{\ell \in \mathcal{T}_0} x_{0i}^{\ell} &= (k_1 - \bar{k}) + \frac{C_i - I_i^0 - Q(\bar{k} - k_1) - u_i t_i^{\bar{k}}}{Q} + \frac{u_i(t_i^{k_2} - t_i^{\bar{k}})}{Q}, \\ &= \frac{C_i - I_i^0}{Q} + \frac{u_i t_i^{k_2}}{Q}, \\ &= \frac{C_i - I_i^0}{Q} + \frac{Hu_i - C_i}{Q}, \\ &= \frac{Hu_i - I_i^0}{Q}.\end{aligned}$$

□

We expect that the previous proposition is true also without specifying a specific number of vehicles, i.e., that the value of the linear relaxation is (4.2) even if the number of vehicles is free and not specified in advance. This proposition shows that the linear relaxation is weak: when the integrality condition on x variables is relaxed, the optimal solution might contain several "fractional" vehicles that deliver only what is consumed in a time interval, $\bar{u}_i^k$, $i \in N$, $k \in \mathcal{T}_i$.

This suggests that strengthening the formulation, by exploiting problem structure, will be critical in the development of an effective solution approach. In the next section we present some observations that can be used for strengthening the formulation (4.1).

4.4 Optimality Preserving Conditions (OPC)

In this section we present several conditions that can be used to limit the search for an optimal solution for the CIPR-OB. We call these conditions *optimality preserving*, and we refer to them as *optimality preserving conditions* (OPC), because there always exists an optimal solution that satisfies *all* these conditions. In order to derive these conditions, we first need to consider the following definitions.

Definition 1. *Depot-time set of a vehicle: For a CIRP-OB feasible solution, we define the depot-time set S_0^v of a vehicle indexed by $v \in \{1, \ldots, m\}$, as the union of time intervals in $[0, H]$ the vehicle v stays at the depot.*

Definition 2. *Depot-time set of a solution: For a CIRP-OB feasible solution, we define the depot-time set of the solution as the collection of depot-time sets of the vehicles, S_0^v, $v \in \{1, \ldots, m\}$.*

Definition 3. *Maximal depot-time-set optimal solution (MDO): A MDO is an optimal solution to a CIRP-OB instance, with depot time set S_0^{v*}, such that there is no other optimal solution, with depot time set S_0^v, $v \in \{1, \ldots, m\}$, that satisfies the following conditions:*

- *for all $v = 1, \ldots, m$, $S_0^{v*} \subseteq S_0^v$;*
- *and there exists a $v' \in \{1, \ldots, m\}$ such that $S_0^{v'*} \subset S_0^{v'}$.*

For the CIRP-OB several similar solutions might be optimal. Small perturbations to an optimal solution in the time a vehicle waits at a customer location, or in the delivered quantity, or in the time a vehicle stays at the depot, lead to different optimal solutions. We restrict the search for optimal solutions to those that are MDO only.

In the following sections we present OPC conditions. Any MDO must satisfy all the OPC conditions. If a solution does not satisfy any of the OPC conditions, then it is not a MDO, since there exists another optimal solution whose depot-time set is maximal. In Section 4.4.1, OPC that restrict the time or the conditions under which vehicles wait at the customer locations are shown. In Section 4.4.2, we present OPC that impose conditions on the visiting times or the delivery quantities when vehicles arrive to a customer.

4.4.1 Vehicle Waiting Conditions

Proposition 19. *In an MDO, if a vehicle waits at customer $i \in N$ before making a delivery, then the inventory level after the delivery is C_i.*

Proof. For contradiction, consider any MDO solution where a vehicle $v \in \{1, \ldots, m\}$ makes a delivery and the inventory level is strictly less than C_i, $i \in N$. Let I_i be the customer inventory level before the delivery. Let $0 < q < C_i - I_i$ be the delivered quantity and let $r > 0$ be the remaining quantity at the vehicle immediately after the delivery. If $I_i + q + r \leq C_i$, then the vehicle can deliver $q + r$ and go back to the depot. This is a contradiction; the solution cost is the same, so it maintains optimality condition, and the depot-time set of the vehicle v, S_0^v, is superset since the vehicle doesn't wait at the customer location. If $I_i + q + r > C_i$, then we can make the vehicle deliver $C_i - I_i$ and wait at the customer location. The remaining quantity at the vehicle is less than r and so the vehicle can return at time $t + \frac{q+r-C_i+I_i}{u_i} < t + \frac{r}{u_i}$, with u_i the customer usage rate and t the vehicle delivery time. The cost is the same and the S_0^v is also superset, so again, this is a contradiction. □

Proposition 20. *In an MDO, if a vehicle waits at customer $i \in N$ before making a delivery, then the inventory level upon arrival is zero.*

Proof. Let $[t_1, t_2]$ be the time interval a vehicle $v \in \{1, \ldots, m\}$ is waiting at the customer location of a MDO solution. Let I_i be the inventory level at time t_1 before delivery and let u_i be the usage rate, $i \in N$. For contradiction, assume I_i is positive. Then, the vehicle can arrive at time $t = \min\{I_i/u_i + t_1, t_2\}$, instead of t_1, and the solution is still feasible. If $t = t_2$, then there is no waiting; otherwise, the vehicle can deliver C_i quantity (the maximum quantity possible to be delivered at time t), and the depot-time set S_0^v is superset, since the vehicle v stays at the depot longer. The solution is feasible and the cost remains the same, reaching the contradiction. □

Proposition 21. *In an MDO, if a vehicle waits at customer $i \in N$ before making a delivery, then the waiting time is no more than $\frac{\max\{Q-C_i,0\}}{u_i}$ time.*

Proof. For contradiction assume that the waiting time of a vehicle $v \in \{1,\ldots,m\}$ is strictly greater than $\frac{\max\{Q-C_i,0\}}{u_i}$ for a MDO solution. By proposition 20, we know the inventory is zero when the vehicle v arrives. If $Q \leq C_i$, the vehicle can deliver Q and immediately after return to the depot. The waiting time at the customer is zero so we get the contradiction. If $Q > C_i$, since the inventory is zero, the vehicle can deliver C_i. Then it can stay until the customer consumes the difference $Q - C_i$ and return to the depot. In this case, the total waiting time is exactly $\frac{Q-C_i}{u_i}$, leading to the contradiction. □

4.4.2 Visit Time Conditions

Let $N_\ell \subseteq N$ be the set of customers whose inventory capacity is less than Q, $C_i < Q$, and let $N_g = N \setminus N_\ell$ be the set of customers whose capacity is greater than or equal to, $C_i \geq Q$.

Proposition 22. *In an MDO, the delivered quantity at the time the vehicle arrives to a customer $i \in N$ is equal to the minimum between $C_i - I_i$ and Q, where I_i is the inventory level at the arrival time and C_i is the customer capacity.*

Proof. Note that if the arriving vehicle waits after the delivery, then by proposition 19, the delivery must be equal to $C_i - I_i$. Consider a vehicle that doesn't wait after the delivery in a MDO solution. If the vehicle delivers the maximum quantity possible, in this case the minimum between the vehicle capacity Q and the remaining customer capacity $C_i - I_i$, then the solution cost is the same and the depot-time set of the solution is at least the same. Indeed, since the maximum possible quantity was delivered by the vehicle, any other vehicle $v \in \{1,\ldots,m\}$ that makes a delivery after can save time at the customer; less product must be delivered. Then, the vehicle v can potentially stay at the depot longer, so the set S_0^v is superset. □

Proposition 23. *In a MDO, if a customer $i \in N_g$ has exactly $\eta_i = \left\lceil \frac{Hu_i - I_i^0}{Q} \right\rceil$ visits, then the visit arrival times s_k, $k = 1 \ldots, \eta_i$ must occur in the following intervals,*

$$s_k \in \left[\frac{Hu_i - C_i - (\eta_i - k)Q}{u_i}, \frac{I_i^0 + (k-1)Q}{u_i} \right], \qquad k = 1, \ldots, \eta_i. \tag{4.3}$$

Proof. First we prove the intervals are correct. We know that $Q\eta_i \geq Hu_i - I_i^0$, so summing kQ both sides with $k = 1, \ldots, \eta_i$, and noticing that $Q \leq C_i$, we get,

$$Hu_i - I_i^0 + kQ \leq Q(\eta_i + k),$$
$$Hu_i - (\eta_i - k)Q \leq I_i^0 + kQ,$$
$$Hu_i - (\eta_i - k)Q - C_i \leq I_i^0 + (k-1)Q.$$

We prove that the visit times must be in the intervals. For contradiction, suppose there is a visit $k = 1, \ldots, \eta_i$ whose time s_k is not in the interval we defined. If the time $s_k > \frac{I_i^0 + (k-1)Q}{u_i}$, then the solution is not feasible since the maximum product delivered by $k-1$ visits is $(k-1)Q$, so at time s_k the inventory level is negative. Then, it must be $s_k < \frac{Hu_i - C_i - (\eta_i - k)Q}{u_i}$. Let q_ℓ be the total delivery done by the vehicle at visit $\ell = 1, \ldots, \eta_i$. The total delivery $\sum_{\ell=1}^{k} q_\ell$ up to time s_k satisfies,

$$\sum_{\ell=1}^{k} q_\ell \leq u_i s_k + C_i - I_i^0,$$

since for customer i no waiting is allowed (proposition 21). The total delivery by all

visits holds,

$$\begin{aligned}
\sum_{\ell=1}^{\eta_i} q_\ell &= \sum_{\ell=1}^{k} q_\ell + \sum_{\ell=k+1}^{\eta_i} q_\ell \\
&\leq \sum_{\ell=1}^{k} q_\ell + (\eta_i - k)Q \\
&\leq u_i s_k + C_i - I_i^0 + (\eta_i - k)Q, \\
&< Hu_i - C_i - (\eta_i - k)Q + C_i - I_i^0 + (\eta_i - k)Q, \\
&= Hu_i - I_i^0,
\end{aligned}$$

but this contradicts the fact the solution is feasible. □

Proposition 24. *In a MDO, if a customer $i \in N_\ell$ has exactly $\eta_i = \left\lceil \frac{Hu_i - I_i^0}{Q} \right\rceil$ visits, then the visit arrival times s_k, $k = 1 \ldots, \eta_i$ must occur in the following intervals,*

$$s_k \in \left[\frac{Hu_i - C_i + I_i^0 - (\eta_i - k + 1)Q}{u_i}, \frac{I_i^0 + (k-1)Q}{u_i} \right], \qquad k = 1, \ldots, \eta_i. \tag{4.4}$$

Proof. This proof is similar to proposition 23's proof.

First we prove the intervals are correct. We know that $Q\eta_i \geq Hu_i - I_i^0$, so summing $(k-1)Q$ both sides with $k = 1, \ldots, \eta_i$, and noticing that $I_i^0 - C_i \leq 0$, we get,

$$\begin{aligned}
Hu_i - I_i^0 + (k-1)Q &\leq Q(\eta_i + k - 1), \\
Hu_i - (\eta_i - k + 1)Q &\leq I_i^0 + (k-1)Q, \\
Hu_i - (\eta_i - k + 1)Q - C_i + I_i^0 &\leq I_i^0 + (k-1)Q.
\end{aligned}$$

We prove that the visit times must be in the intervals. For contradiction, suppose there is a visit $k = 1, \ldots, \eta_i$ whose time s_k is not in the interval we defined. The case $s_k > \frac{I_i^0 + (k-1)Q}{u_i}$ is equivalent to 23's proof. Let q_ℓ be the total delivery done by the vehicle at visit $\ell = 1, \ldots, \eta_i$. Suppose the time s_k is before $\frac{Hu_i - C_i + I_i^0 - (\eta_i - k + 1)Q}{u_i}$. There are two cases, the vehicle during the visit kth waits or it doesn't. If it waits,

then the inventory level before the delivery at the arrival time is zero (proposition 20), so it must be $\sum_{\ell=1}^{k-1} q_\ell = u_i s_k - I_i^0$. In this case, the total delivery up to time s_k, $\sum_{\ell=1}^{k} q_\ell$, is at most $u_i s_k - I_i^0 + Q$, then,

$$\begin{aligned}
\sum_{\ell=1}^{\eta_i} q_\ell &\leq u_i s_k - I_i^0 + Q + (\eta_i - k)Q, \\
&< Hu_i - C_i + I_i^0 - (\eta_i - k + 1)Q - I_i^0 + Q + (\eta_i - k)Q, \\
&= Hu_i - C_i, \\
&\leq Hu_i - I_i^0,
\end{aligned}$$

which is a contradiction.

Let I_i^k be the inventory intermediately before the kth delivery. If the vehicle doesn't wait during the kth visit then, the inventory level after the delivery at time s_k is C_i (proposition 22), so the q_k delivery is $C_i - I_i^k$ and also,

$$\begin{aligned}
\sum_{\ell=1}^{k} q_\ell &= \sum_{\ell=1}^{k-1} q_\ell + q_k, \\
&= I_i^k - I_i^0 + s_k u_i + C_i - I_i^k, \\
&= s_k u_i + C_i - I_i^0.
\end{aligned}$$

Thus, the total delivery satisfies,

$$\begin{aligned}
\sum_{\ell=1}^{\eta_i} q_\ell &\leq s_k u_i + C_i - I_i^0 + (\eta_i - k)Q, \\
&< Hu_i - C_i + I_i^0 - (\eta_i - k + 1)Q + C_i - I_i^0 + (\eta_i - k)Q, \\
&= Hu_i - Q, \\
&< Hu_i - C_i, \\
&\leq Hu_i - I_i^0.
\end{aligned}$$

leading to the contradiction. □

Proposition 25. *In a MDO, if a customer $i \in N_g$ has v visits, $v > \eta_i = \left\lceil \frac{Hu_i - I_i^0}{Q} \right\rceil$, then in at least one of the intervals,*

$$\left[\frac{kQ - (C_i - I_i^0)}{u_i}, \frac{I_i^0 + (k-1)Q}{u_i} \right], \qquad k = 1, \ldots, \eta_i, \tag{4.5}$$

there are no arrivals.

Proof. For contradiction, assume the solution is MDO and all intervals have at least one visit. Consider the following procedure: for each interval keep the latest visit only by removing all the visits before within the same interval. Since each interval $k = 1, \ldots, \eta_i$ is defined from time $s_1^k = \frac{kQ - (C_i - I_i^0)}{u_i}$, the visit we left is guaranteed to delivery Q product. Indeed, the visit time t is greater than or equal to s_1^k thus the delivery q_k in this visit is the minimum between the capacity Q and the difference between the inventory before the visit and the capacity C_i,

$$\begin{aligned} q_k &= \min\{C_i - (tu_i - I_i^0 - (k-1)Q), Q\}, \\ &\geq \min\{C_i - (kQ - (C_i - I_i^0) - I_i^0 - (k-1)Q), Q\}, \\ &= Q. \end{aligned}$$

For each interval k the inventory is non-negative at the moment of the vehicle arrival t. Indeed, for each $\ell = 1, \ldots, k-1$ the delivery is Q, then the inventory is non-negative at $t \in [s_1^k, s_2^k]$, with $s_2^k = \frac{I_i^0 + (k-1)Q}{u_i}$,

$$\begin{aligned} I_i^0 + (k-1)Q - tu_i &\geq I_i^0 + (k-1)Q - s_2^k, \\ &\geq I_i^0 + (k-1)Q - I_i^0 - (k-1)Q, \\ &\geq 0. \end{aligned}$$

We conclude the solution we propose is feasible and its cost is less than the MDO

solution cost, reaching the contradiction. □

4.5 Lower Bound Model (LBM)

A formulation that provides a lower bound on the value of an optimal solution to an instance of CIRP-OB can be obtained from formulation (4.1) by removing time points from the discretization, i.e., removing one or more time points from the sets $\mathcal{T}_i$ for $i \in N$, appropriately adjusting input parameters, and relaxing some of the constraints (the time expanded network must satisfy $\{0, H\} \subseteq \{t_i^k\}_{k \in \mathcal{T}_i}$ for all $i \in N_0$). More specifically, consider the following modifications:

1. The travel time at time t_i^k, $i \in N_0$, $k \in \mathcal{T}_i$, to a location $j \in N_0$ is given according to the following expression,

$$\tau_{ij}^k = \begin{cases} \max_{\ell \in \mathcal{T}_j}\{t_j^\ell : t_j^\ell \leq t_0^k + \tau_j\} - t_0^k & i = 0, j \in N, \\ \max_{\ell \in \mathcal{T}_0}\{t_0^\ell : t_0^\ell \leq t_i^k + \tau_i\} - t_i^k & i \in N, j = 0, \\ t_i^{k+1} - t_i^k & i = j, k < K_i. \end{cases} \tag{4.6}$$

   The travel times are rounded down, which ensures that $\tau_{ij}^k \leq \tau_j$, for all $i, j \in N_0$, $k \in \mathcal{T}_i$. Note that, depending on the time discretizations, some travel times τ_{ij}^k might be non-positive. Also for two different departure time points $t_i^{k_1}$ and $t_i^{k_2}$, $k_1, k_2 \in \mathcal{T}_i$, with $k_1 < k_2$, the arrival time at $j \in N_0$ can be potentially the same, i.e., $t_i^{k_1} + \tau_{ij}^{k_1} = t_i^{k_2} + \tau_{ij}^{k_2}$.

2. The customer storage capacity is enlarged by adding the amount $u_i(t_i^{k+1} - t_i^k) = \bar{u}_i^{k+1}$, i.e., the product consumption during the time interval $[t_i^k, t_i^{k+1}]$, with $i \in N$, $k \in \mathcal{T}_i$. We modify the z variables upper bound in (4.1i) as follows,

$$\bar{u}_i^{k+1} \leq z_i^k \leq C_i + \bar{u}_i^{k+1}. \tag{4.7}$$

Note that the new storage capacity bounds depend on the customer time discretization; different time interval lengths have different inventory capacities.

3. Multiple vehicles can visit a customer at the same time. We remove one-visit-at-the-time constraints (4.1e) and we also let the variables x_{ij}^k be defined for any non-negative integer, $x_{ij}^k \in \mathbb{Z}_+$, $a_{ij}^{k\ell} \in \mathcal{A}^{\mathcal{T}}$.

We call this model the Lower Bound Model (LBM), which is justified by the following theorem.

Theorem 2. *The optimal value of the LBM is a lower bound on the optimal value of the CIRP-OB.*

Proof. Consider any CIRP-OB feasible solution and any time discretization. Let $\{\upsilon_i^\ell\}_{\ell \in L_i}$ be the sequence of decision times, namely, arrival to; departure from; or delivery time of location $i \in N_0$, with set index L_i. Similarly, let η_i^ℓ be the delivered quantity to customer $i \in N$ at time υ_i^ℓ, $\ell \in L_i$. We show that this CIRP-OB solution can be mapped to a LBM solution at no extra cost. Consider the following mapping of a time υ_i^ℓ to a LBM time $T_i(\upsilon_i^\ell)$,

$$T_i(\upsilon_i^\ell) = \max_{k \in \mathcal{T}_i} \left\{ t_i^k : t_i^k \leq \upsilon_i^\ell \right\}.$$

Every time point υ_i^ℓ is mapped to the closest time t_i^k in the time discretization, with $t_i^k \leq \upsilon_i^\ell$, $i \in N_0$.

The travel times defined in (4.6) guarantee the LBM solution is feasible in time dimension, i.e., the vehicle flow balance is preserved. Let $\upsilon_0^{\ell_1}$ be any departure time from the depot to some customer $i \in N$, $\ell_1 \in L_0$, and let $\upsilon_i^{\ell_2}$ be the corresponding arrival time to i, $\ell_2 \in L_i$ in the CIRP-OB solution. Since the solution is feasible, it holds $\upsilon_0^{\ell_1} + \tau_i \leq \upsilon_i^{\ell_2}$. Let $k_1 \in \mathcal{T}_0$ and $k_2 \in \mathcal{T}_i$ such that $T_0(\upsilon_0^{\ell_1}) = t_0^{k_1}$ and $T_i(\upsilon_i^{\ell_2}) = t_i^{k_2}$,

then,

$$\begin{aligned} t_0^{k_1} + \tau_{0i}^{k_1} &= \max_{k \in \mathcal{T}_i} \{ t_i^k : t_i^k \leq t_0^{k_1} + \tau_i \} \\ &\leq \max_{k \in \mathcal{T}_i} \left\{ t_i^k : t_i^k \leq \upsilon_0^{\ell_1} + \tau_i \right\} \\ &\leq \max_{k \in \mathcal{T}_i} \left\{ t_i^k : t_i^k \leq \upsilon_i^{\ell_2} \right\} \\ &= t_i^{k_2}. \end{aligned}$$

An equivalent argument can be used to show that departing from a customer $i \in N$ location to the depot in the LBM is time feasible.

If a vehicle is waiting at the customer $i \in N$ location, then for any two consecutive times $\upsilon_i^{\ell_1}$ and $\upsilon_i^{\ell_2}$ in the same vehicle itinerary, it holds either $t_i^{k_1} = T_i(\upsilon_i^{\ell_1}) = T_i(\upsilon_i^{\ell_2}) = t_i^{k_2}$ and there is no waiting in the LBM solution; or $t_i^{k_2} = t_i^{k_2} + \sum_{k=k_1}^{k_2-1} \tau_{ii}^k$.

So any decision time in the CIRP-OB solution is mapped to a time in the LBM solution that is before in the time horizon. Since in the LBM waiting times at customer place and more than one visit at the same time are allowed, the itinerary given by the LBM times is feasible.

The LBM solution keeps the deliveries η_i^ℓ values, for all $i \in N$ and $\ell \in L_i$. The CIRP-OB solution is product flow balanced, so we only show that the LBM solution is feasible for inventory levels. Let $I_i(t)$ be the inventory level at time $t \in [0, H]$ in the CIRP-OB solution, for a customer $i \in N$. Note that $0 \leq I_i(t) \leq C_i$, for all $t \in [0, H]$. Consider any integer $k < K_i$ and the interval given by $[t_i^k, t_i^{k+1}]$. Let υ_i^ℓ be the time of the last visit to customer i in that interval, $\ell \in L_i$. Since the CIRP-OB solution is feasible it must be that $I(\upsilon_i^\ell) \geq (t_i^{k+1} - \upsilon_i^\ell) u_i$. It also holds $t_i^k = T(\upsilon_i^\ell)$. It follows,

$$z_i^k = I_i^0 + \sum_{s=1}^{\ell} \eta_i^s - u_i t_i^k = I_i^0 + \sum_{s=1}^{\ell} \eta_i^s - u_i \upsilon_i^\ell + u_i(\upsilon_i^\ell - t_i^k) = I_i(\upsilon_i^\ell) + u_i(\upsilon_i^\ell - t_i^k),$$

Combining the above, we get,

$$z_i^k = I_i(v_i^t) + u_i(v_i^t - t_i^k) \geq (t_i^{k+1} - v_i^\ell)u_i + u_i(v_i^t - t_i^k) = u_i(t_i^{k+1} - t_i^k) = \bar{u}_i^{k+1},$$

and also,

$$z_i^k \leq C_i + u_i(v_i^t - t_i^k) \leq C_i + \bar{u}_i^{k+1}.$$

Thus, the inventory levels are feasible.

We conclude, the CIRP-OB solution can be mapped to a LBM solution with no extra cost. In particular, this is true for an optimal CIRP-OB solution, so the LBM is a valid lower bound model. □

Figure 4.1 shows an example of a mapping of travel times in the LBM. Black dots represent time points for the customer $i \in N$ and the depot time discretization. The solid line represents a CIRP-OB solution and the crosses are times at which the customer is visited in that solution. This solution can be mapped as it is shown with dashed lines. Note that in this mapping the vehicle in the LBM has to wait at the customer location in order to maintain the CIRP-OB itinerary, even when the vehicle in the CIRP-OB solution does not.

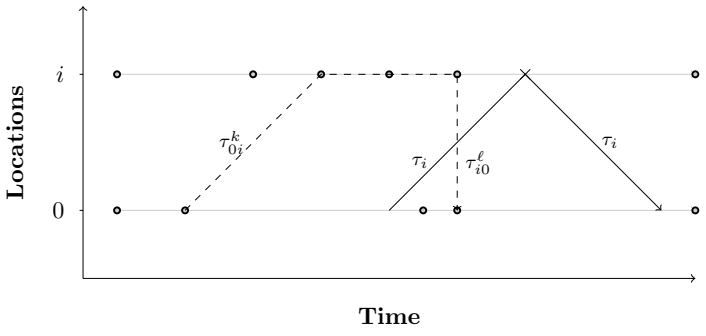

Figure 4.1: LBM Travel times example

Figure 4.2 shows an example of customer inventory level in a LBM solution mapped from a CIRP-OB solution. The solid line represents the inventory level

for a feasible CIRP-OB solution whose delivery times are v and delivery quantities are η. In this solution the inventory levels are always non-negative and below capacity C_i. The mapped LBM solution is represented with a dashed line, with deliveries at times t. The inventory levels in the LBM are above C_i at times t_i^{k-1} and t_i^k in order to deliver the same CIRP-OB amount.

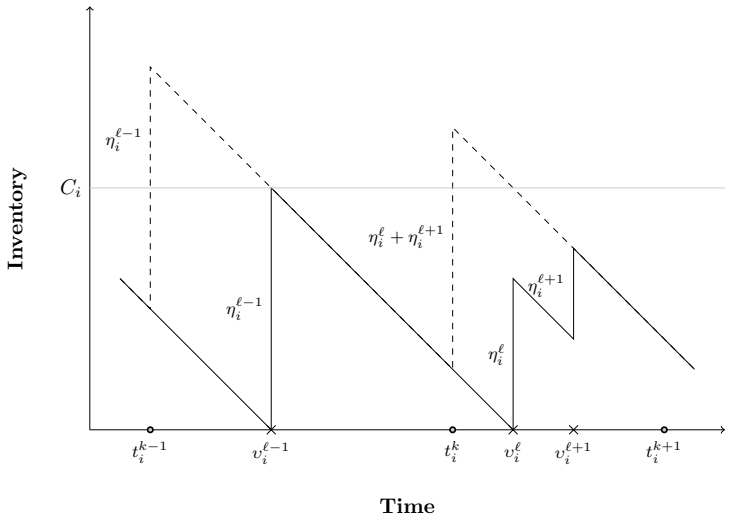

Figure 4.2: Example of a LBM inventory levels

As mentioned above, the travel times are rounded down, which means that it is possible that for two different time points in the departure location, the arrival time at the destination is the same. This condition gives more symmetry to the LBM, making it more difficult to solve in practice. We can remove some of the x variables that are redundant in the LBM, as stated in the following proposition.

Proposition 26. *In the LBM, if two points* $(0, t_0^{\ell_1})$ *and* $(0, t_0^{\ell_2})$ *in* $\mathcal{N}^{\mathcal{T}}$*, with* $t_0^{\ell_1} < t_0^{\ell_2}$ *have a timed arc connecting the same customer* $i \in N$ *at the same time* t_i^k*,* $k \in \mathcal{T}_i$*, then the variable* x *associated to the point* $(0, t_0^{\ell_1})$ *can be removed from the formulation.*

Proof. This result follows from the observation that the LBM without the time point $(0, t_0^{\ell_1})$ is still a lower bound model for the formulation, since the mapping T_i, $i \in N$, in proposition 17 considers the closest time point in the time-expanded network for the CIPR-OB solution. □

In the LBM the OPC conditions cannot be imposed directly. However, we can check if the time-expanded network satisfies some conditions that allows to include the OPC in the model.

4.5.1 Incorporating Vehicle Waiting Conditions into the LBM

In this section, we show how the conditions of Section 4.4.1 can be enforced by constraints in the LBM. We call a LMDO to a solution in the LBM for which there exists a MDO that can be mapped to the LBM solution.

Proposition 27. *(Condition Proposition 19): Any LMDO satisfies the following constraints,*

$$\begin{aligned} & C_i\nu_i^k \leq z_i^k \leq (C_i + \bar{u}_i^{k+1}) - \bar{u}_i^{k+1}\nu_i^k, && i \in N, k \in \mathcal{T}_i, \\ & x_{ii}^k \leq m\nu_i^k, && i \in N, k \in \mathcal{T}_i, \\ & \nu_i^k \in \{0,1\}, && i \in N, k \in \mathcal{T}_i, \end{aligned} \tag{4.8}$$

provided $\bar{u}_i^{k+1} \leq C_i$. The binary variable ν_i^k is equal to one if at least one vehicle is waiting at the customer $i \in N$ at time t_i^k, $k \in \mathcal{T}_i$; zero otherwise.

Proof. Let's first consider the case with a single vehicle waiting at a customer location $i \in N$. Then at some time t_i^k, $k \in \mathcal{T}_i$, we have $x_{ii}^k = 1$. This means that for any CIRP-OB feasible solution mapped to this LBM solution there must be a vehicle waiting from some time t in $[t_i^k, t_i^{k+1})$ to some time t' in $[t_i^{k+1}, t_i^{k+2})$. Given that $\bar{u}_i^{k+1} \leq C_i$, we can assume w.l.o.g. that there are at most two deliveries in $[t, t_i^{k+1}]$, one at time t and another at time t_i^{k+1} (we can consolidate small deliveries in that interval). The proposition 19 holds that at any time after a delivery the inventory level must be equal to C_i, in particular, this is true at t. In the LBM, the time t maps to t_i^k so the inventory at this time must be equal to C_i.

If $x_{ii}^k > 1$ for the LBM solution, then more than vehicle is waiting at the customer

$i \in N$ location, but there is at most one doing deliveries, since the LBM is mapped from a feasible CIRP-OB solution and at most one vehicle can deliver at the same time. For the vehicle that is doing deliveries, using the previous result, every time x_{ii}^k is positive, the inventory must be equal to C_i. □

Proposition 28. *If for an index $k \in \mathcal{T}_i$ there exists a $\ell \in \mathcal{T}_0$, such that $t_0^\ell + \tau_{0i}^\ell = t_i^k$, then any LMDO satisfies the following constraints,*

$$x_{ii}^{k-1} \leq 1, \qquad \text{if } i \in N_\ell,$$
$$x_{ii}^{k-1} = 0, \qquad \text{if } i \in N_g.$$

Proof. Consider a customer $i \in N_\ell$. Every CIRP-OB feasible solution satisfies that no more than one vehicle is waiting at some customer location. If the LBM solution mapped from this solution has more than one vehicles are waiting at i at time t_i^{k-1}, $k \in \mathcal{T}_i$, means that at least $v-1$, with $v = x_{ii}^{k-1} > 1$, vehicles make a delivery at some time in $[t_i^k, H]$ and no delivery before time t_i^k. Assume those $v-1$ vehicles depart at time $t_0^{\ell'}$ from the depot, then we can re-route them from time $t_0^{\ell'}$ to time t_0^ℓ, arriving at time t_i^k to customer i. The time t_0^ℓ and travel time τ_{0i}^ℓ guarantee this new itinerary is feasible. We keep the same itinerary after this time t_i^k. Therefore the constraint $x_{ii}^{k-1} \leq 1$ is valid.

Consider a customer $i \in N_g$. A CIRP-OB feasible solution satisfies that waiting vehicles is not allow for i. Similar to the case above, if $v = x_{ii}^{k-1} > 0$ vehicles are waiting we can re-route from time $t_0^{\ell'}$ to time t_0^ℓ, arriving at time t_i^k to customer i, so the constraint $x_{ii}^{k-1} = 0$ is valid. □

Proposition 29. *(Condition Proposition 20): If for some customer $i \in N$ and $k \in \mathcal{T}_i$ there exist indexes $\ell_1, \ell_2 \in \mathcal{T}_0$ such that $t_0^{\ell_1} + \tau_{0i}^{\ell_1} = t_i^k$ and $t_0^{\ell_2} + \tau_{0i}^{\ell_2} = t_i^{k+1}$, then any LMDO satisfies the following constraint,*

$$(C_i - \bar{u}_i^k)(2 - x_{i0}^{\ell_1} - x_{ii}^k) + \bar{u}_i^{k+1} \geq z_i^{k-1} - \bar{u}_i^k. \tag{4.9}$$

Proof. From Proposition 28, we know that the number of vehicles waiting from some time in $[t_i^k, t_i^{k+1})$ to some time in $[t_i^{k+1}, t_i^{k+2})$ is at most one, since the depot time discretization has a time $t_0^{\ell_2}$ such that $t_0^{\ell_2} + \tau_{0i}^{\ell_2} = t_i^{k+1}$. Consider any optimal solution to the CIPR-OB, then by proposition 20, any vehicle that arrives at some time in $[t_i^k, t_i^{k+1})$ and then waits, must get there when the customer is zero level inventory. This means that any LBM optimal solution in which a vehicle visits a customer at time t_i^k and then waits, the customer can have at most $\bar{u}_i^{k+1}$ units in inventory. □

Proposition 30. *(Condition Proposition 21): Consider a customer $i \in N_\ell$. For an index $k_1 \leq K_i - 1$ and an index $k_2 = \text{argmin}_{k \in \mathcal{T}_i}\{t_i^k \leq t_i^{k_1+1} + \frac{Q-C_i}{u_i}\}$ such that there exists a $\ell \in \mathcal{T}_0$, with $t_0^\ell + \tau_{0i}^\ell = t_i^{k_2}$. There exist a LMDO that satisfies the following constraint,*

$$\sum_{k=k_1}^{k_2} x_{ii}^k \leq k_2 - k_1.$$

Proof. For any arrival time during the interval $[t_i^{k_1}, t_i^{k_1+1})$ the departure time cannot be later than $t_i^{k_1+1} + (Q - C_i)/u_i$ since the waiting time is at most $(Q - C_i)/u_i$ for customers $i \in N_\ell$ (Proposition 21). Thus, a vehicle can wait at most from time $t_i^{k_1}$ to $t_i^{k_2}$ in the LBM. □

4.5.2 Incorporating Visit Time Conditions into the LBM

Consider the condition given in Proposition 22: the delivery at time t_i^k must be the minimum of Q and $C_i - I_i^k$, where I_i^k is the inventory level before the delivery, for any customer $i \in N$. Let v_i^k be a binary variable that is equal to one if the minimum of the vehicle capacity Q and $C_i - I_i^k$ is Q; zero otherwise; for a customer $i \in N_g$ and a time t_i^k, $k \in k \in \mathcal{T}_i$. Any LMDO satisfies the following constraints,

$$y_i^k \geq (C_i + \bar{u}_i^k)(x_{0i}^\ell - v_i^k) - z_i^{k-1}, \tag{4.10a}$$

$$y_i^k \geq Q(x_{0i}^\ell + v_i^k - 1). \tag{4.10b}$$

For customers $i \in N_\ell$, the delivery is always $C_i - I_i^k$, so the following constraint is satisfied by any LMDO at a time t_i^k, $k \in k \in \mathcal{T}_i$,

$$y_i^k \geq (C_i + \bar{u}_i^k)x_{0i}^\ell - z_i^{k-1}, \qquad i \in N_\ell, k \in \mathcal{T}_i. \tag{4.11}$$

Note that $I_i^k = C_i + \bar{u}_i^k - z_i^{k-1}$, so when $x_{0i}^\ell - v_i^k = 1$ for customers $i \in N_g$, or when $x_{0i}^\ell = 1$ for customers $i \in N_\ell$, the inventory level after the delivery is C_i.

Inputs for the inequalities in this section are a lower and an upper bound for the number of visits to a customer. The lower bound η_i can be obtained computing the total product must be delivered $Hu_i - I_i^0$ divided by the vehicle capacity Q (maximum product can be carried by a vehicle), $\eta_i = \left\lceil \frac{Hu_i - I_i^0}{Q} \right\rceil$, for all $i \in N$. An upper bound M_i for the number of visits can be computed as the maximum total time that can be assigned to customer i divided by the total time it takes to visit it, $2\tau_i$. We have,

$$M_i = \left\lfloor \frac{mH - \sum_{j \in N: j \neq i} 2\tau_j \eta_j}{2\tau_i} \right\rfloor, \qquad i \in N.$$

For customers $i \in N$ with storage capacity $C_i > Q$, a system of inequalities can be derived from Propositions 23 and 25 for the LBM. Consider the following time indexes:

$$\begin{aligned}
s_1^k &= \max\{k \in \mathcal{T}_i : u_i t_i^k \leq Hu_i - C_i - (\eta_i - k)Q\} && k = 1, \ldots, \eta_i, \\
s_2^k &= \max\{k \in \mathcal{T}_i : u_i t_i^k \leq I_i^0 + (k-1)Q\} && k = 1, \ldots, \eta_i, \\
s_3^k &= \max\{k \in \mathcal{T}_i : u_i t_i^k \leq kQ - C_i + I_i^0\} && k = 1, \ldots, \eta_i, \\
s_4^k &= \max\{k \in \mathcal{T}_i : u_i t_i^k \leq I_i^0 + (k-1)Q - \epsilon\} && k = 1, \ldots, \eta_i,
\end{aligned}$$

with $0 < \epsilon \leq C_i - Q$, given. Note that s_1^k is the index in $\mathcal{T}_i$ that represents the lower bound in the interval in (4.3) for the kth visit arrival time in the LBM, while s_2^k is

the index that represents the upper bound in the interval. The index s_3 is the time index for the lower bound in the interval in (4.5) in the LBM and the s_4^k represents the upper bound of that interval.

Let ν_i a binary variable that is equal to one if the number of visits to customer i is η_i; zero otherwise. Let r_i^k be a binary variable that is equal to one if the interval given by $[t_i^{s_3^k}, t_i^{s_4^k}]$ has at least one visit; zero otherwise; for $k = 1, \ldots, \eta_i$. Any LMDO satisfies the following system,

$$(\eta_i + 1)(1 - \nu_i) \leq \sum_{\ell \in \mathcal{T}_0} x_{0i}^{\ell} \leq M_i - (M_i - \eta_i)\nu_i, \tag{4.12a}$$

$$\nu_i \leq \sum_{s=s_1^k}^{s_2^k} \sum_{\ell:(0,\ell)\in\delta^-(i,t_i^s)} x_{0i}^{\ell}, \qquad k = 1, \ldots, \eta_i \tag{4.12b}$$

$$\sum_{s=s_3^k}^{s_4^k} \sum_{\ell:(0,\ell)\in\delta^-(i,t_i^s)} x_{0i}^{\ell} \leq M_i r_i^k, \qquad k = 1, \ldots, \eta_i \tag{4.12c}$$

$$\sum_{k=1}^{\eta} r_i^k \leq \eta - 1 + \nu_i. \tag{4.12d}$$

provided the following conditions are satisfied in the time-expanded network,

1. there exists a index $\ell \in \mathcal{T}_0$ such that $t_0^{\ell} + \tau_{0i}^{\ell} = t_i^{s_1^k}$,
2. there exists a index $\ell \in \mathcal{T}_0$ such that $t_0^{\ell} + \tau_{0i}^{\ell} = t_i^{s_3^k}$.

The two conditions, time points in $\{t_0^{\ell}\}_{\ell \in \mathcal{T}_0}$ that permit to get to i at time $t_i^{s_1^k}$ and at time $t_i^{s_3^k}$, guarantee that a vehicle arriving at those times is not represented with a waiting arc in the LBM.

For customers $i \in N$ with storage capacity $C_i \leq Q$, a system of inequalities can be derived from Proposition 24. As before, we define time indexes representing the

interval in (4.4) in the LBM:

$$s_1^k = \max\{k \in \mathcal{T}_i : u_i t_i^k \leq H u_i - C_i + I_i^0 - (\eta - k + 1)Q\} \qquad k = 1, \ldots, \eta_i,$$
$$s_2^k = \max\{k \in \mathcal{T}_i : u_i t_i^k \leq I_i^0 + (k-1)Q\} \qquad k = 1, \ldots, \eta_i.$$

Let ν_i be a binary variable that is equal to one if the number of visits to customer i is η_i; zero otherwise. Any LMDO satisfies the following system,

$$(\eta_i + 1)(1 - \nu_i) \leq \sum_{\ell \in \mathcal{T}_0} x_{0i}^\ell \leq M_i - (M_i - \eta_i)\nu_i, \tag{4.13a}$$

$$\nu_i \leq \sum_{s=s_1^k}^{s_2^k} \sum_{\ell:(0,\ell)\in\delta^-(i,t_i^s)} x_{0i}^\ell, \qquad k = 1, \ldots, \eta_i, \tag{4.13b}$$

provided there exists a index $\ell \in \mathcal{T}_0$ such that $t_0^\ell + \tau_{0i}^\ell = t_i^{s_1^k}$, so it is guaranteed the arrival time at $t_i^{s_1^k}$ is represented with a direct arc in the LBM.

4.5.3 Valid Inequalities for the Number of Visits

In [72] the authors present several valid inequalities for the CIRP. A large class of them corresponds to lower bounds on the number of visits to a customer, adaptations of inequalities presented in [67] and [71]. In our setting, we keep the inequalities in [67] and we adapt them to the CIPR-OB. We consider a lower bound on the number of visits to a customer that must occur in an arbitrary time interval.

From a time $t_1 \in \{t_i^k\}_{k\in\mathcal{T}_i}$ and to a time $t_2 \in \{t_i^k\}_{k\in\mathcal{T}_i}$ with $t_1 < t_2$, the customer $i \in N$ consumption in that interval is $u_i(t_2 - t_1)$. At time t_1 the customer has at most C_i product in inventory, thus during $[t_1, t_2]$ the customer requires at least $u_i(t_2 - t_1) - C_i$ to be delivered. Since the maximum product that can be delivered during a visit is Q, we can compute a lower bound for the number of visits to i.

Proposition 31. *The following inequalities are valid for the LBM formulation,*

$$\left\lceil \frac{u_i(t_i^{k_2+1} - t_i^{k_1}) - C_i}{Q} \right\rceil \leq \sum_{k=k_1}^{k_2} \sum_{(0,\ell)\in\delta^-(i,t_i^k)} x_{0i}^{\ell}, \qquad i \in N_g, k_1 \in \mathcal{T}_i, k_1 < k_2, \tag{4.14}$$

$$\left\lceil \frac{u_i(t_i^{k_2+1} - t_i^{k_1}) - C_i}{Q} \right\rceil \leq x_{ii}^{k_1-1} + \sum_{k=k_1}^{k_2} \sum_{(0,\ell)\in\delta^-(i,t_i^k)} x_{0i}^{\ell}, \qquad i \in N_\ell, k_1 \in \mathcal{T}_i, k_1 < k_2. \tag{4.15}$$

Proof. For a customer $i \in N$, let k_1 and k_2 be any two indexes in $\mathcal{T}_i$ such that $k_1 < k_2$. At time $t_i^{k_2}$, we know the inventory level is bounded from below as follows in the the LBM,

$$z_i^{k_2} \geq \bar{u}_i^{k_2+1} = u_i(t_i^{k_2+1} - t_i^{k_2}).$$

Also, summing up the inventory balance constraints from time $t_i^{k_1}$ to $t_i^{k_2}$, we have,

$$z_i^{k_2} = z_i^{k_1-1} + \sum_{k=k_1}^{k_2} y_i^k - u_i(t_i^{k_2} - t_i^{k_1-1}),$$

so combining both inequalities we get,

$$\sum_{k=k_1}^{k_2} y_i^k \geq u_i(t_i^{k_2+1} - t_i^{k_1-1}) - z_i^{k_1-1},$$

and noticing that at time $t_i^{k_1-1}$ the inventory level is bounded from above by $C_i + u_i(t_i^{k_1} - t_i^{k_1-1})$, we finally have,

$$\sum_{k=k_1}^{k_2} y_i^k \geq u_i(t_i^{k_2+1} - t_i^{k_1-1}) - (C_i + u_i(t_i^{k_1} - t_i^{k_1-1})),$$
$$\sum_{k=k_1}^{k_2} y_i^k \geq u_i(t_i^{k_2+1} - t_i^{k_1}) - C_i,$$

which is a valid inequality for any customer $i \in N$. We can use this expression in order to get (4.14) and (4.15) by deriving an inequality for y and x variables. For

customers $i \in N_g$, there is no waiting, so all the product that gets to a customer must have arrival time at the interval $[t_i^{k_1}, t_i^{k_2}]$, so,

$$\sum_{k=k_1}^{k_2} y_i^k = \sum_{k=k_1}^{k_2} \sum_{(0,\ell)\in\delta^-(i,t_i^k)} w_{i0}^\ell \leq Q \sum_{k=k_1}^{k_2} \sum_{(0,\ell)\in\delta^-(i,t_i^k)} x_{i0}^\ell,$$

then we have,

$$\frac{u_i(t_i^{k_2+1} - t_i^{k_1}) - C_i}{Q} \leq \sum_{k=k_1}^{k_2} \sum_{(0,\ell)\in\delta^-(i,t_i^k)} x_{i0}^\ell.$$

For customers $i \in N_\ell$, waiting is allowed, so some product can come from a vehicle waiting from the previous interval. We have,

$$\sum_{k=k_1}^{k_2} y_i^k = w_{ii}^{k_1-1} - w_{ii}^{k_2} + \sum_{k=k_1}^{k_2} \sum_{(0,\ell)\in\delta^-(i,t_i^k)} w_{i0}^\ell \leq Q x_{ii}^{k_1-1} + Q \sum_{k=k_1}^{k_2} \sum_{(0,\ell)\in\delta^-(i,t_i^k)} x_{i0}^\ell,$$

so we conclude,

$$\frac{u_i(t_i^{k_2+1} - t_i^{k_1}) - C_i}{Q} \leq x_{ii}^{k_1-1} + \sum_{k=k_1}^{k_2} \sum_{(0,\ell)\in\delta^-(i,t_i^k)} x_{i0}^\ell.$$

□

Note that the previous inequalities are valid for any $k_1 < k_2$, with $k_1, k_2 \in \mathcal{T}_i$ and $i \in N$, but not all valid inequalities for pairs of time indexes are useful in the LBM. If the value of the lhs for a given interval is less or equal to zero, the inequality does not strengthen the model. Thus, we only add inequalities with $u_i(t_i^{k_2+1} - t_i^{k_1}) - C_i > 0$.

4.5.4 Branch-and-Bound Strategy

Consider the example in Figure 4.3. For this instance there is one customer c with usage rate $u_c = 1$, storage capacity $C_c = 10$ and initial inventory $I_c^0 = 10$. The

time horizon is $H = 11$, so the customer needs at least one visit and the vehicle must deliver at least one unit of product. The vehicle capacity is $Q = 10$. The cost and travel times are equal, both with value 1. Assume that the time discretization for the customer and the depot is the set $\{1, 2, \ldots, 11\}$. For this instance and time discretization is easy to check that there exists an optimal solution for formulation (4.1).

If we solve the linear relaxation of the formulation, the solution has a fraction 1/10 of a vehicle, that delivers 1 unit of product at time $t_c^1 = 1$. In order to get an integer solution, we consider the following branching scheme. In one branch there is a complete vehicle ($x_{0i}^0 = 1$) and the solution is integer. In the other branch, no vehicle departs at time t_0^0 from the depot ($x_{0i}^0 = 0$) and another fraction solution is optimal. A 2/10 fractional vehicle arrives at time 2 and delivers 2 units of product (note that 2 units are delivered because of the OPC in Proposition 22: the inventory after the delivery must be C_i). Again, it is needed to branch this solution: one branch finds the optimal solution ($x_{0i}^1 = 1$), while another has a 3/10 fractional vehicle arriving at time $t = 3$. Continuing in this fashion, after ten branchings a provable optimal solution is found.

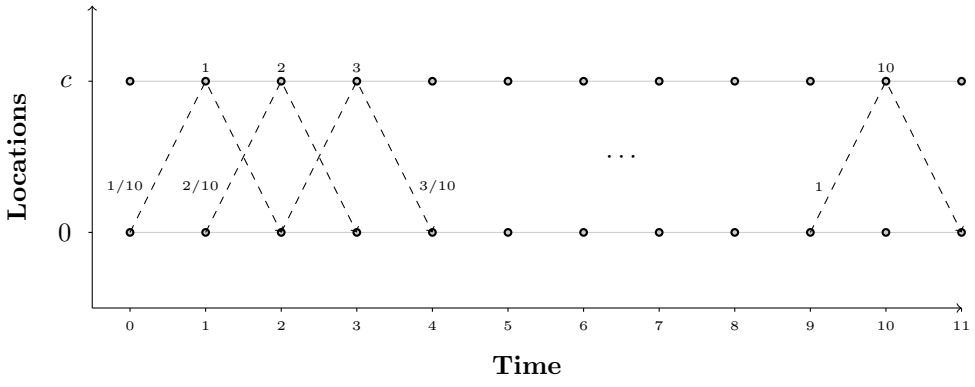

Figure 4.3: Simple Branch-and-Bound for a one-customer instance

To avoid this behavior, we seek a branching scheme that results in a more balanced search tree. We consider the variables x_{0i}^ℓ that represent visits to a customer $i \in N$ at

time $k \in \mathcal{T}_i$, with $\ell \in \delta^-(i, t_i^k)$. In the branching strategy we propose, the first value we branch on is the total number of visits to i, i.e., we branch on the following sum,

$$S_i^{00} = \sum_{k=0}^{K_i} \sum_{\ell \in \delta^-(i,t_i^k)} x_{i0}^{\ell}.$$

When this sum is integer, the next step is branching on the following two sums,

$$S_i^{10} = \sum_{k=0}^{\lceil \frac{K_i}{2} \rceil - 1} \sum_{\ell \in \delta^-(i,t_i^k)} x_{i0}^{\ell},$$

$$S_i^{11} = \sum_{k=\lceil \frac{K_i}{2} \rceil}^{K_i} \sum_{\ell \in \delta^-(i,t_i^k)} x_{i0}^{\ell}.$$

Note that the value S_i^{00} is the sum of the values S_i^{10} and S_i^{11}. Once the two sums S_i^1 for this level have integer value, we continue branching on the following S_i^2 level. In general, for a level n, the sum $S_i^{n,k}$, $k = 0, \ldots, 2^n - 1$, corresponds to the following expression,

$$S_i^{n,k} = \sum_{k'=k_1}^{k_2} \sum_{\ell \in \delta^-(i,t_i^{k'})} x_{i0}^{\ell}, \tag{4.16}$$

with $k_1 = k \left\lceil \frac{K_i}{2^n} \right\rceil$ and $k_2 = \min \left\{ (k+1) \left\lceil \frac{K_i}{2^n} \right\rceil - 1, K_i \right\}$.

For the last level n, each value $S_i^{n,k}$ satisfies $k_2 - k_1 \leq 1$. Thus, the number of levels is $n = \lceil \log_2(K_i) \rceil - 1$.

The branching rule we propose starts branching on S_i^0 at the first level and, once it has integer value, the branching rule continues with S_i^1. Then, it branches on S_i^2, S_i^3, and so on, up to level $\lceil \log_2(K_i) \rceil - 1$. The level n is successfully branched when, for each customer $i \in N$, the sums S_i^n, that define the number of visits for a particular interval, have an integer value.

4.5.5 Recovering Customer Storage Capacity Constraints

In order to get an LBM from the formulation (4.1), the travel times, customer storage capacities and number of visits at the same time time are modified. If the time-expended network is fine enough, the travel times will be correct and the number of visits can be handled (as shown in section 4.7.3), but the customer inventory bounds will be always $C_i + \bar{u}_i^k$, with $\bar{u}_i^k > 0$ for all $i \in N$, $k \in \mathcal{T}_i$. Even if the number of time points $\{t_i^k\}_{k\in\mathcal{T}_i}$ is large and they are no more than $\epsilon > 0$ apart, $\max_{k\in\mathcal{T}_i}\{t_i^{k+1} - t_i^k\} \leq \epsilon$, the inventory bound is strictly greater than C_i. How do we know LBM provides an useful solution for the problem? Theorem 3 states that for a fine time discretization the solution for x variables is the same for the LBM formulation with $C_i + \bar{u}_i^k$ and for the LBM with C_i.

The analysis in this section assumes the time discretization for any location $i \in N_0$ is Δ-homogeneous, i.e., for all $k \in \mathcal{T}_i$, $t_i^{k+1} - t_i^k = \Delta$. For notation, we represent by LBM$^+(\Delta)$ the LBM over a Δ-homogeneous time discretization and customer inventory capacity bounded by $C_i + u_i\Delta$. Similarly, the LBM(Δ) is the the LBM over a Δ-homogeneous time discretization and customer inventory capacity bounded by C_i.

Consider an optimal solution for the LBM$^+(\Delta)$. Let $\omega_{i,a}^k(\Delta)$ be the kth arrival time to location $i \in N_0$ in the LBM solution, with $k = 1, \ldots, \eta_i$, and η_i the number of visits to i in the solution. Equivalently, let $\omega_{i,d}^k(\Delta)$ the kth departure time from the location $i \in N_0$, with $k = 1, \ldots, \eta_i$.

Consider an optimal solution for the LBM(Δ). Let $v_{i,a}^k(\Delta)$ be the kth arrival time to $i \in N_0$, and let $v_{i,d}^k(\Delta)$ be the kth departure time from location i of the optimal LBM solution, $k = 1, \ldots, \eta_i$.

Theorem 3. *There exist a $\Delta > 0$ and arrival and departure times $\omega_{i,a}^k(\Delta)$, $\omega_{i,d}^k(\Delta)$, $v_{i,a}^k(\Delta)$, $v_{i,d}^k(\Delta)$, with the same number of visits η_i, such that $\omega_{i,a}^k(\Delta) = v_{i,a}^k(\Delta)$ and $\omega_{i,d}^k(\Delta) = v_{i,d}^k(\Delta)$, for all $i \in N_0$ and $k = 1, \ldots, \eta_i$.*

Proof. For contradiction, we assume the following,

> **(A)**: *There is no* $\Delta > 0$ *and there is no times* $\omega_{i,j}^k(\Delta)$ *and* $\upsilon_{i,j}^k(\Delta)$ *such that the number of visits* η_i *is the same and* $\omega_{i,j}^k(\Delta) = \upsilon_{i,j}^k(\Delta)$, *for all* $i \in N_0$, $j = a, d$, $k = 1, \ldots, \eta_i$.

Take any $\Delta > 0$ such that for all $i \in N$, $\tau_i/\Delta \in \mathbb{Z}$ and $H/\Delta \in \mathbb{Z}$ (data is rational, so there is a Δ that satisfies those conditions). Let $\Delta_\ell = \Delta 2^{-\ell}$, $\ell \geq 0$, be an infinite sequence. Note that for all $\ell \geq 0$, in the LBM$^+(\Delta_\ell)$, the number of visits to a customer $i \in N$ is bounded. Indeed, since for all $\ell \geq 0$, it also holds $\tau_i/\Delta_\ell \in \mathbb{Z}$, $H/\Delta_\ell \in \mathbb{Z}$, then $\frac{\lceil H/\Delta_\ell \rceil}{2\lfloor \tau_i/\Delta_\ell \rfloor} \leq \frac{H}{2\tau_i}$ for all $i \in N$.

Let $\eta(\Delta_\ell) = [\eta_1(\Delta_\ell), \ldots, \eta_{|N|}(\Delta_\ell)]$ be a vector of number of visits to customers of an optimal solution to LBM$^+(\Delta_\ell)$, $\ell \geq 0$. Since the number of visits to each customer is bounded, the set $\{\eta(\Delta_\ell)\}_{\ell \in \mathcal{L}}$ is finite. Also, there are a finite number of feasible itineraries (i.e., the assignment of those visits to vehicles) for those visits. Then, we can take an infinite subsequence $\{\Delta_k\}_{k \in \mathcal{K}}$, with $\mathcal{K} \subseteq \mathbb{Z}_+$, such that there is an optimal solution to the LBM$^+(\Delta_k)$ whose number of visits is the same $\bar{\eta} = \eta(\Delta_k)$ and those visits occur in the same sequence, for all $k \in \mathcal{K}$. For each LBM$^+(\Delta_k)$, $k \in \mathcal{K}$, we consider an optimal solution (only one) whose number of visits $\bar{\eta}$ and the itineraries are the same, so we refer to *the* optimal solution to LBM$^+(\Delta_k)$.

Note that the number of visits given by $\bar{\eta}$ does not induce a feasible solution to LBM(Δ'), for all $\Delta' > 0$, because any feasible solution to the LBM with inventory capacity C_i is a feasible solution to the LBM with inventory capacity $C_i + u_i\Delta'$. This contradicts **(A)**, since the number of visits is optimal for both LBM(Δ') and LBM$^+(\Delta')$ and the visit arrival and departure times are the same.

Let $f_i^\ell(\Delta_k)$ be the total delivery to $i \in N$ during the ℓth visit of the LBM$^+(\Delta_k)$ optimal solution, $k \in \mathcal{K}$, $\ell = 1, \ldots, \bar{\eta}_i$. Similarly, let $z_i^\ell(\Delta_k)$ be the inventory level

before the delivery ℓth. Define,

$$\bar{f}_i^\ell = \lim_{k\to\infty} f_i^\ell(\Delta_k),$$
$$\bar{z}_i^\ell = \lim_{k\to\infty} z_i^\ell(\Delta_k).$$

Note that $\bar{f}_i^\ell$ and $\bar{z}_i^\ell$ represent feasible deliveries and inventory levels (in the limit there is no extra inventory capacity).

We also define $\bar{v}_i^\ell = \lim_{k\to\infty} v_{i1}^\ell(\Delta_k)$, and $\bar{w}_i^\ell = \lim_{k\to\infty} v_{i2}^\ell(\Delta_k)$, $\ell = 1,\ldots,\bar{\eta}_i$, $i \in N$.

There must be a $i \in N$ and a $\ell = 1,\ldots,\bar{\eta}_i$ such that either $\bar{v}_i^\ell$ or $\bar{w}_i^\ell$ is irrational. If not, then all arrival and departure times are rational, so there must exist a $\Delta' > 0$ such that Δ' is divisor for all times. The solution $(\bar{f}_i^\ell, \bar{z}_i^\ell, \bar{v}_i^\ell, \bar{w}_i^\ell)$ represents a feasible solution to LBM(Δ'), but this is a contradiction: we know there is no feasible solution to LBM(Δ') with number of visits given by $\bar{\eta}$, for all $\Delta' > 0$.

Using the number of visits $\bar{\eta}$ and the itineraries of LBM$^+(\Delta_k)$ optimal solutions, $k \in \mathcal{K}$, we can construct a polytope. Let v_i^ℓ be the ℓth visit arrival time and let w_i^ℓ be the ℓth visit departure time, $i \in N$. Let z_i^ℓ be the inventory level immediately before the ℓth visit and let f_i^ℓ the total delivery made by the ℓth visit, $i \in N$. Let $s(i,\ell) = (j,k) \in N \times \{1,\ldots,\bar{\eta}_j\}$ be the customer j and the visit kth that goes after the ℓth visit to i in the same vehicle itinerary, $i \in N$, $\ell = 1,\ldots,\bar{\eta}_i$. The polytope for

$(v_i^\ell, w_i^\ell, f_i^\ell, z_i^\ell)$ variables is as follows,

$$w_i^{\ell-1} \leq v_i^\ell, \qquad i \in N, \ell = 2, \ldots, \bar{\eta}_i, \tag{4.17a}$$

$$z_i^1 = I_i^0 - u_i v_i^1, \qquad i \in N, \tag{4.17b}$$

$$z_i^\ell = z_i^{\ell-1} + f_i^\ell - u_i(v_i^\ell - v_i^{\ell-1}), \qquad i \in N, \ell = 2, \ldots, \bar{\eta}_i, \tag{4.17c}$$

$$z_i^{\bar{\eta}_i} + f_i^{\bar{\eta}_i} \geq u_i(H - v_i^{\bar{\eta}_i}), \qquad i \in N, \tag{4.17d}$$

$$z_i^\ell + f_i^\ell \leq C_i + u_i(w_i^\ell - v_i^\ell), \qquad i \in N, \ell = 2, \ldots, \bar{\eta}_i, \tag{4.17e}$$

$$v_i^\ell \geq w_j^k + \tau_i + \tau_j, \qquad i \in N, \ell = 2, \ldots, \bar{\eta}_i, (j,k) = s(i,\ell), \tag{4.17f}$$

$$\tau_i \leq v_i^\ell \leq w_i^\ell \leq H - \tau_i, \qquad i \in N, \ell = 2, \ldots, \bar{\eta}_i, \tag{4.17g}$$

$$0 \leq f_i^\ell \leq Q, \qquad i \in N, \ell = 2, \ldots, \bar{\eta}_i, \tag{4.17h}$$

$$z_i^\ell \geq 0, \qquad i \in N, \ell = 2, \ldots, \bar{\eta}_i.$$

Constraints (4.17a) ensure that the visit times at customers occur sequentially. Constraints (4.17b) and (4.17c) set the inventory levels just prior to each visit at a customer. Constraints (4.17d) ensure that inventory after the last delivery is sufficient to meet demand until the end of the planning horizon. Constraints (4.17e) enforce that the vehicle remains long enough at the customer to deliver its whole load, while not violating the capacity limit. Constraints (4.17f) ensure that for each of the vehicles the visit times at customers properly account for travel times between locations, and, in case a vehicle performs multiple routes, properly account for travel times to and from the depot in between consecutive routes in its itinerary. Constraints (4.17g) impose bounds on arrival and departure times. Finally, constraints (4.17h) enforce deliveries cannot be more than the vehicle capacity Q.

Note that there cannot be a rational solution for the previous polytope (4.17) since that solution would be feasible for LBM(Δ'), for some $\Delta' > 0$, which again, contradicts **(A)**. However, the solution $(\bar{f}_i^\ell, \bar{z}_i^\ell, \bar{v}_i^\ell, \bar{w}_i^\ell)$ is a feasible solution to that

polytope, so it is not empty. Therefore, a non-empty polytope with rational data does not have a rational point (solution), reaching a contradiction.

□

4.6 Finding Feasible Solutions for the CIRP-OB

In this section we present two different methods to get feasible solutions for a CIPR-OB instance. First, we consider a MIP formulation that, if feasible, provides feasible solutions for the continuous time problem for any time discretization. This formulation is derived in a similar way to LBM: the formulation (4.1) is changed and then we show the solution we get from the new formulation has a transportation cost that is an upper bound for the problem. Then, we present a MIP formulation that finds continuous visit times and deliveries for an optimal number of visits from the LBM; thus, we can determine whether a LBM optimal solution is feasible for the CIPR-OB or not.

4.6.1 Upper Bound Model (UBM)

An upper bound formulation for the CIRP-OB can be derived from formulation (4.1). The model we propose contains the same variables and constraints defined for formulation (4.1), but the travel times are modified: they are rounded up in the time-expanded network. The travel time at a time t_i^k, $i \in N_0$, $k \in \mathcal{T}_i$, to a location $j \in N_0$, is given according to the following expression,

$$\tau_{ij}^k = \begin{cases} \min_{\ell \in \mathcal{T}_j}\{t_j^\ell : t_j^\ell \geq t_0^k + \tau_j\} - t_0^k, & i = 0, j \in N, t_0^k + \tau_j \leq H \\ \min_{\ell \in \mathcal{T}_0}\{t_0^\ell : t_0^\ell \geq t_i^k + \tau_i\} - t_i^k, & i \in N, j = 0, t_i^k + \tau_i \leq H \\ t_i^{k+1} - t_i^k & i = j, k < K_i. \end{cases} \tag{4.18}$$

We call this model the Upper Bound Model (UBM). Even though the formulation

decision times are restricted to the location time discretizations, the UBM finds feasible solutions for the CIRP-OB. Therefore, this model is a valid upper bound model for the problem.

Proposition 32. *Any feasible solution to the UBM is a feasible solution to the CIRP-OB.*

Proof. The UBM travel times ensure that a feasible solution for the model is feasible for the continuous time problem. Any vehicle whose travel times are given by (4.18), it gets to the destination after a vehicle traveling according to τ_i, $i \in N$. Note also that the continuous time solution can keep the UBM solution customers itinerary: if the continuous time vehicle gets too early to the customer, it can wait longer at the depot before departure. The constraints in formulation (4.1) ensure the solution is feasible for the CIRP-OB. □

Even if a feasible solution to an instance of CIRP-OB exists, UBM may not find it. The travel times and the constraints of the formulation might produce an infeasible model. For a Δ-homogeneous time discretization for the UBM, i.e., time intervals such that $t_i^{k+1} - t_i^k = \Delta > 0$, for all $i \in N_0$ and $k \in \mathcal{T}_i$, we show conditions on Δ that ensure a feasible model.

Proposition 33. *If* $\Delta > 0$ *satisfies,*

$$u_i \leq \frac{I_i^0}{\Delta\left(\left\lceil \frac{C_i}{Q} \right\rceil + \left\lceil \frac{\tau_i}{\Delta} \right\rceil\right)}, \tag{4.19}$$

$$\left\lceil \frac{C_i}{Q} \right\rceil \leq \frac{H}{\Delta} - 2\left\lceil \frac{\tau_i}{\Delta} \right\rceil, \tag{4.20}$$

for all $i \in N$*, then the UBM with unlimited number of vehicles is feasible using a* Δ*-homogeneous time discretization.*

Proof. We prove that the following solution is feasible: visit each customer $i \in N$ at any possible time delivering as much as possible. Since the number of vehicles is

unrestricted, we need to check if the inventory level are non-negative during the all time horizon for all customers. In particular, we show that for all $i \in N$ the inventory level is non-negative at time $\Delta\lceil I_i^0/(\Delta u_i)\rceil$ and the inventory is enough to cover the time $H - \Delta\lceil \tau_i/\Delta\rceil$ to return to the depot.

For each $i \in N$, the first visit time in the solution is at Δk_1, with $k_1 = \lceil \tau_i/\Delta\rceil$ and the last visit time is at Δk_2, with $k_2 = \lfloor H/\Delta\rfloor - k_1$. Note that (4.20) guarantees that,

$$k_2 - k_1 \geq \left\lceil \frac{C_i}{Q} \right\rceil.$$

Also, at time Δk with $k = \left\lfloor \frac{I_i^0}{u_i\Delta} \right\rfloor$, the inventory level z_i^k is equal to C_i since the total delivery up to k is at least $k\Delta u_i$. Indeed, using (4.19) we have,

$$k - k_1 = \left\lfloor \frac{I_i^0}{u_i\Delta} \right\rfloor - \left\lceil \frac{\tau_i}{\Delta} \right\rceil \geq \left\lceil \frac{C_i}{Q} \right\rceil,$$

so,

$$(k - k_1)Q \geq C_i \geq k\Delta u_i.$$

The inventory level after time $k\Delta$ is also C_i and the delivery of each visit is at most Q,

$$\Delta u_i \leq \frac{I_i^0}{\left\lceil \frac{C_i}{Q} \right\rceil + \left\lceil \frac{\tau_i}{\Delta} \right\rceil} \leq \frac{C_i}{\left\lceil \frac{C_i}{Q} \right\rceil + \left\lceil \frac{\tau_i}{\Delta} \right\rceil} < \frac{C_i}{\left\lceil \frac{C_i}{Q} \right\rceil} \leq Q.$$

Finally, at time Δk_2 the inventory is C_i and it is enough to cover the customer

usage up to time H,

$$\begin{aligned} H - \Delta k_2 &\leq \Delta + \left\lceil \frac{\tau_i}{\Delta} \right\rceil \Delta, \\ &\leq \Delta \left(\left\lceil \frac{C_i}{Q} \right\rceil + \left\lceil \frac{\tau_i}{\Delta} \right\rceil \right), \\ &\leq \frac{I_i^0}{u_i}, \\ &\leq \frac{C_i}{u_i}, \end{aligned}$$

so the product consumption while the last vehicle returns to the depot satisfies $u_i(H - \Delta k_2) \leq C_i$. □

4.6.2 Feasibility Check Model (FCM)

An optimal solution to LBM specifies a number of visits to each customer. These visits have a total transportation cost that is a lower bound for the CIRP-OB, but it is unknown whether this number of visits leads to a solution that is feasible for the continuous time problem or not.

In this section we present a model that allows us to determine whether a continuous-time feasible solution with this number of visits exists. We formulate a model that takes as an input the number of visits n_i, $i \in N$, and finds new continuous visiting times, defined in the interval $[0, H]$, and delivered quantities, defined in $[0, Q]$. These times and quantities must satisfy the CIRP-OB constraints, including the conditions we relax for the LBM: the travel times are given by the parameter τ_i, $i \in N$; the customer inventory capacities are bounded by C_i, $i \in N$; and the number of vehicles waiting at the same time at the customer location is at most one. We construct a mixed-integer programming (MIP) model to decide (revise) the visiting times and delivered quantities, while preserving the number of visits to each customer. We call this formulation Feasibility Check Model (FCM). Naturally, it may be that no feasible CIRP-OB solution using these number of visits exists, in which case a new optimal LBM solution is needed.

Each visit to a customer is associated to an out-and-back route. Let $R = \{1, \ldots, \sum_{i \in N} n_i\}$ be the index set of routes. For each customer $i \in N$, let $R_i \subseteq R$ be the set of routes that visit customer $i \in N$. Let $c(r) \in N$ denote the customer visited on route $r \in R$. We assume customers have at least one visit, $n_i \geq 1$ and, w.l.o.g., the routes in R_i visit customer i in route index order, $R_i = \{r_1^i, r_2^i, \ldots, r_{n_i}^i\}$ with $r_{k-1}^i < r_k^i$ for all $k = 2, \ldots, n_i$. We have $c(r_k^i) = i$ for all $k = 1, \ldots, n_i$.

Let variable v_r be the arrival time at customer $c(r)$ and let w_r be the departure time from customer $c(r)$ of route $r \in R$. Inventory variables, z_r, denote the inventory level at customer $c(r)$ immediately prior to the first delivery on route r. Let y_{r_1,r_2} be a binary variable that is one if the same vehicle does route r and then r': zero otherwise. The variable ζ is the minimum slack time between any two consecutive visits.

$$
\begin{aligned}
\max \quad & \zeta, \\
\text{s.t.} \quad & \zeta \leq v_{r_k^i} - w_{r_{k-1}^i}, && i \in N, k = 2, \ldots, n_i, && \text{(4.21a)} \\
& z_{r_1^i} = I_i^0 - u_i v_{r_1^i}, && i \in N, && \text{(4.21b)} \\
& z_{r_k^i} = z_{r_{k-1}^i} + f_{r_{k-1}^i} - u_i(v_{r_k^i} - v_{r_{k-1}^i}), && i \in N, k = 2, \ldots, n_i, && \text{(4.21c)} \\
& z_{r_{n_i}^i} + f_{r_{n_i}^i} \geq u_i(H - v_{r_{n_i}^i}), && i \in N, && \text{(4.21d)} \\
& z_r + f_r \leq C_{c(r)} + u_{c(r)}(w_r - v_r), && r \in R, && \text{(4.21e)} \\
& v_{r'} \geq w_r + \tau_{c(r)} + \tau_{c(r')} - M(1 - y_{r,r'}), && r, r' \in R && \text{(4.21f)} \\
& \tau_{c(r)} \leq v_r \leq w_r \leq H - \tau_{c(r)}, && r, r' \in R, r \neq r' && \text{(4.21g)} \\
& w_r = v_r, && r \in R, c(r) \in N_g && \text{(4.21h)} \\
& \sum_{r' \in R} y_{r,r'} \leq 1, && r \in R && \text{(4.21i)} \\
& \sum_{r' \in R} y_{r',r} \leq 1, && r \in R && \text{(4.21j)} \\
& \sum_{r \in R} \sum_{r' \in R} y_{r,r'} \geq |R| - m && && \text{(4.21k)} \\
& 0 \leq f_r \leq Q, && r \in R && \text{(4.21l)} \\
& y_{r,r'} \in \{0, 1\}, && r, r' \in R, && \\
& z_r \geq 0, && r \in R. &&
\end{aligned}
$$

Constraints (4.21a) ensure that the visit times at customers occur sequentially with at least ζ units of time apart. Constraints (4.21b) and (4.21c) set the inventory levels just prior to each visit at a customer. Constraints (4.21d) ensure that inventory after the last delivery is sufficient to meet demand until the end of the planning horizon. Constraints (4.21e) enforce that the vehicle remains long enough at the customer to deliver its whole load, while not violating the capacity limit. Constraints (4.21f) ensure that for each of the vehicles the visit times at customers properly account for

travel times between locations, and, in case a vehicle performs multiple routes, properly account for travel times to and from the depot in between consecutive routes in its itinerary. Constraints (4.21g) impose bounds on arrival and departure times. Note that the w_r variables are only needed if $c(r) \in N_\ell$, but for simplicity of exposition, we include them for all r, and impose the constraints (4.21h) in the model. Constraints (4.21i)-(4.21k) ensure a sequence of routes for each vehicle itinerary that satisfies the vehicle availability m. Finally, constraints (4.21l) enforce deliveries cannot be more than the vehicle capacity Q.

If the FCM is feasible and has optimal value $\zeta^* > 0$, then the solution provides a feasible solution to the CIRP-OB with vehicle movement cost the same as the cost of the LBM solution. If no feasible solution with positive ζ exists in the FCM, then for the number of visits n_i given as an input, $i \in N$, there are no visit times and deliver quantities that result in a feasible CIRP-OB solution.

In order to guarantee the LBM solution is CIRP-OB feasible, we only need to find a feasible solution to the FCM with $\zeta > 0$. In our implementation we start with the sequence provided by the LBM solution for the FCM model (initial solution for binary y variables) and we run the solver until a solution with $\zeta > 0$ is found.

In practice, we have seen this model solves very fast, even for large instances it takes no more than few seconds. For the instances we present, there is no need of including OPC into the FCM.

4.7 Dynamic Discovery Discretization Algorithm (DDD) for the CIRP-OB

The DDD algorithm is presented in [8] for the continuous time service network design problem. The central idea of the algorithm methodology is to work with a partial time discretization, that is sequentially and precisely refined, so it is guaranteed an optimal continuous time solution can be produced. This algorithm consists of the

following steps:

1. find a lower bound solution for the continuous time problem in a time-expanded network;

2. given a lower bound solution, determine whether it is feasible for the (original) problem or can be converted into a feasible solution for the (original) problem (which implies it is optimal);

3. if the solution cannot be converted, improve the LBM by adding time points to the partial discretization.

In this section we develop the full algorithm for the CIRP-OB. Steps 1 (LBM) and 2 (FCM) have been already presented in previous sections, so we describe Step 3 and the execution of the full algorithm.

With respect to the CIRP-OB, the LBM considers three relaxations: travel times are rounded down; inventory capacity is enlarged by adding the consumption of the next time interval; and multiple visits to a customer at the same time is allowed. After checking that an optimal solution for the LBM is not feasible for the CIRP-OB problem, we detect one or more of the mentioned relaxations to correct in the time-expanded network, so the current optimal solution is no longer feasible for the LBM.

4.7.1 Correcting Travel Times

Since the travel times in the LBM are shorter than τ_i, $i \in N$, some difficulties can arise when trying to convert the LBM solution into a feasible continuous time solution: it is not possible to get to a customer before it runs out of product since the out-and-back route duration is too long, or the vehicle itinerary is larger than the total time H, or both.

Algorithm 3 takes as input the optimal LBM times at which vehicles depart from a location. For each $x_{ij}^k > 0$, $a_{ij}^{k\ell} \in \mathcal{A}^{\mathcal{T}}$, this algorithm checks whether the travel times starting at time t_i^k are correct. If a shorter travel time to the depot, $j = 0$, is detected, the algorithms adds the point $t_i^k + \tau_i$ to the set $\{t_0^\ell\}_\ell$, provided the new time point satisfies $t_i^k + \tau_i \leq H$. Equivalently, if a shorter travel time from the depot, $i = 0$, is detected, the time point $t_i^k + \tau_j$ is added to the set $\{t_j^\ell\}_\ell$, provided $t_i^k + \tau_j \leq H$.

```
1  forall x_{i0}^k > 0 do
2  |  set s_1 = t_i^k + τ_i;
3  |  if s_1 ∉ {t_0^ℓ}_{ℓ∈T_0} and s_1 ≤ H then
4  |  |  add s_1 to {t_0^ℓ}_{ℓ∈T_0};
5  |  end
6  end
7  forall x_{0j}^k > 0 do
8  |  set s_2 = t_0^k + τ_j;
9  |  if s_2 ∉ {t_j^ℓ}_{ℓ∈T_j} and s_2 ≤ H then
10 |  |  add s_2 to {t_j^ℓ}_{ℓ∈T_j};
11 |  end
12 end
```

Algorithm 3: Travel time correcting algorithm.

This algorithm refines the time-expanded network in order to get a LBM solution whose travel times are not rounded down.

4.7.2 Correcting Customer Storage Capacities

Increasing the customer storage capacity in the LBM implies that vehicles can deliver more than is possible when the inventory capacity is C_i, $i \in N$. More product can be accommodated at the customer storage, so fewer out-and-back routes are necessary to supply customers. Also, if a vehicle is waiting in the LBM solution, it can return

to the depot before it is possible in continuous time since it doesn't have to wait the time needed for the delivery.

Algorithm 4 corrects the customer storage capacities. This algorithm checks whether customer storage capacity C_i is being violated by vehicles deliveries in the LBM solution, $i \in N$. If the inventory z_i^k, $k \in \mathcal{T}_i$, exceeds C_i for a large amount, then the time interval given by $[t_i^k, t_i^{k+1}]$ is too long and a new time point is needed to reduce it. A new time point in the middle of that interval, i.e., the time point $\frac{t_i^{k+1}+t_i^k}{2}$ is added to the set $\{t_i^k\}_k$. Since in the LBM the customer capacities are always larger than C_i, we consider a $\epsilon > 0$ tolerance. If $[t_i^k, t_i^{k+1}] \leq \epsilon$, this interval is not split by adding more time points.

input: $\epsilon > 0$

forall $i \in N$ **do**
  forall $k \in \mathcal{T}_i$ **do**
    if $C_i < z_i^k$ *and* $t_i^{k+1} - t_i^k > \epsilon$ **then**
      add time point $\frac{t_i^{k+1}+t_i^k}{2}$ to the set $\{t_i^l\}_l$;
    end
  end
end

Algorithm 4: Inventory capacity correcting algorithm.

Lemma 5. *For a given $\epsilon > 0$, the maximum number of points added by Algorithm 4 is $|N|\lceil 2H/\epsilon \rceil$.*

Proof. We show that in a interval $[t, t+\epsilon/2]$, $t \in [0, H-\epsilon/2]$, at most one point is added by the algorithm. Suppose, for contradiction, in that interval there are two points $t_1, t_2 \in [t, t+\epsilon/2]$, $t_1 < t_2$ and at least one of them is added by the algorithm. If t_1 is in the interval and then t_2 is included by the algorithm, then there exists a t_3 such that $t_2 = t_1 + (t_3 - t_1)/2 \leq t_1 + \epsilon/2$, so $t_3 - t_1 \leq \epsilon$. But $t_3 - t_1 > \epsilon$, otherwise the algorithm

doesn't consider that interval, reaching the contradiction. If t_2 is in the interval and then t_1 is added, then there exists a t_3 such that $t_1 = t_3 + (t_2 - t_3)/2 \geq t_2 - \epsilon/2$, so $t_2 - t_3 \leq \epsilon$. But $t_2 - t_3 > \epsilon$, so we reach the contradiction.

We conclude by observing the maximum number of intervals of length $\epsilon/2$ for a given location $i \in N$ is $\lceil 2H/\epsilon \rceil$. □

4.7.3 Correcting Number of Visits

In the CIRP-OB vehicles are allowed to wait at the customer location but there must be at most one vehicle at the same time. In the LBM this constraint is relaxed. Therefore, if the solutions we get from the LBM are not continuous time feasible, then one possible problem is given by multiple vehicles waiting at a customer location at the same time.

We consider a $\epsilon > 0$ and a LBM solution as an input. Let $\mathcal{S}_i$ be the set of indexes for customer $i \in N$ whose times t_i^k, $k \in \mathcal{S}_i$, have more than one visit at the same time. For all time points t_i^k, $k \in \mathcal{S}_i$ we add the time $t = t_i^k + \epsilon$, provided $t_i^{k+1} > t$ and $t \leq H$. Since at time t_i^k there might be a vehicle that is waiting to do a delivery at time t_i^{k+1} or after, the travel times must also be corrected. We add the time $t - \tau_i$ to $\{t_0^\ell\}_{\ell \in \mathcal{T}_0}$, provided $t - \tau_i \geq 0$.

The algorithm for correcting multiple visits at the customer is shown in Algorithm 5.

input: $\epsilon > 0$

detect time points $\{t_i^k\}_{i \in N, k \in \mathcal{S}_i}$ with multiple visits;

forall $\{t_i^k\}_{i \in N, k \in \mathcal{S}_i}$ **do**

    add one-visit-at-the-time constraint for t_i^k;

    add time $t_i^k + \epsilon$ to $\{t_i^\ell\}_{\mathcal{T}_i}$;

    add time $t_i^k + \epsilon - \tau_i$ to $\{t_0^\ell\}_{\mathcal{T}_0}$;

end

Algorithm 5: Correcting multiple visits at the same time algorithm.

The following lemma and proposition state Algorithm 5 only adds a finite number of time points and also preserves the valid lower bound condition of the LBM.

Lemma 6. *For a given $\epsilon > 0$, the maximum number of points added by Algorithm 5 is $2|N|\lceil 2H/\epsilon \rceil$.*

Proof. Note that for any customer $i \in N$ and any time interval $[t, t + \epsilon/2]$, with $t \in [0, H - \epsilon]$ at most one time point can be added by the algorithm, so no more than $2\lceil 2H/\epsilon \rceil$ points are added to i. Since for each time $t + \epsilon$ added to i there is a time $t + \epsilon - \tau_i$ added to the depot discretization, the algorithm generates at most $2|N|\lceil 2H/\epsilon \rceil$ time points. □

Proposition 34. *There exists a $\epsilon > 0$ such that Algorithm 5 preserves the valid lower bound condition of the LBM.*

Proof. In [72] the authors prove that for any feasible instance of the CIRP there exists a solution with visiting times and quantities rational. Since the CIRP-OB it is a particular case, there exists a rational solution for this problem. In particular, there exists a $\epsilon > 0$ such that all visit times are at least ϵ apart. Thus, if the time-expanded network is dense enough, such that for all $i \in N$ and $k \in \mathcal{T}_i$, $t_i^{k+1} - t_i^k \leq \epsilon$, imposing the one-visit-at-the-time constraint at each $(i, t_i^k) \in \mathcal{N}^{\mathcal{T}}$ preserves at least one optimal solution for the problem. □

4.7.4 DDD algorithm for the CIRP-OB

In this section we describe the DDD algorithm for the CIRP-OB, present properties and guarantees of the algorithm and mention some implementation details. The algorithm uses the correcting algorithms described in Sections 4.7.1, 4.7.2 and 4.7.3. The DDD algorithm is shown in Algorithm 6.

input: $\epsilon > 0$

1 set a initial time discretization $\{t_i^k\}_k$, for all $i \in N_0$;
2 solve the LBM, get optimal number of visits;
3 solve FCM with LBM number of visits;
4 **if** *FCM is feasible and* $\zeta^* > 0$ **then**
5 | return optimal solution;
6 **else**
7 | add time points using *travel time correcting algorithm 3*;
8 | add time points using *inventory capacity correcting algorithm 4* with ϵ;
9 | add time points and constraints using *visits correcting algorithm 5* with ϵ;
10 **end**
11 **if** *no time points are added to the time-expanded network* **then**
12 | stop;
13 **end**
14 go to step 2;

Algorithm 6: DDD algorithm for the CIRP-OB

Algorithm 6 receives a $\epsilon > 0$ tolerance and it starts with a initial time-expanded network. It solves the LBM formulation and gets an optimal number of visits to each customer. Then, the LBM solution is checked using the FCM model, so it is

determined whether it can be converted to a CIRP-OB feasible solution or not. If it can, then the FCM solution is optimal. If it not, then the time-expanded network and the LBM must be improved. The travel time, inventory capacity and visits correcting algorithms are executed. The inventory and visits correcting algorithm are executed using a ϵ-tolerance, i.e., time points are added if the time intervals are greater than ϵ. If at least one time point is added to the time-expanded network in steps 7, 8 and 9, the LBM is solved in the new network. If no points are added, then the algorithm stops.

The algorithm starts with time discretizations $\{t_i^k\}_{\mathcal{T}_i}$, $i \in N$, such that for times 0 and H are contained in all sets, $\{0, H\} \subseteq \{t_i^k\}_{\mathcal{T}_i}$. We also include time points needed for the visit time conditions in Section 4.5.2.

For all $i \in N_g$, the set $\{t_i^k\}_{\mathcal{T}_i}$ has the following time points:

- $s_{i,1}^k = H - \frac{C_i+(\eta_i-k)Q}{u_i}$, for all $k = 1, \ldots, \eta_i$;
- $s_{i,2}^k = \frac{I_i^0+(k-1)Q}{u_i}$, for all $k = 1, \ldots, \eta_i$;
- $s_{i,3}^k = \frac{kQ-C_i+I_i^0}{u_i}$, for all $k = 1, \ldots, \eta_i$;
- $s_{i,4}^k = \frac{I_i^0+(k-1)Q}{u_i} - \epsilon$, for all $k = 1, \ldots, \eta_i$.

For all $i \in N_\ell$, the set $\{t_i^k\}_{\mathcal{T}_i}$ has the following time points:

- $s_{i,1}^k = H - \frac{C_i-I_i^0+(\eta_i-k+1)Q}{u_i}$, for all $k = 1, \ldots, \eta_i$;
- $s_{i,2}^k = \frac{I_i^0+(k-1)Q}{u_i}$, for all $k = 1, \ldots, \eta_i$;

For the depot, the set $\{t_0^k\}_{\mathcal{T}_0}$ has the following time points:

- $s_{0,i,1}^k = s_{i,1}^k - \tau_i$, for all $i \in N$ and $k = 1, \ldots, \eta_i$;
- $s_{0,i,2}^k = s_{i,2}^k - \tau_i$, for all $i \in N$ and $k = 1, \ldots, \eta_i$;

Theorem 4 below states that the DDD algorithm is an exact algorithm for the CIRP-OB, so for a feasible instance with rational data, it finds an optimal solution. Before proving that theorem we need the following lemma and proposition.

Proposition 35. *For a given $\epsilon > 0$, Algorithm 6 finishes after a finite number of steps.*

Proof. We prove that for any $\epsilon > 0$ tolerance, a finite number of time points are added by algorithms 3, 4 and 5, so the algorithm finites in a finite number of iterations. We assume the initial time discretization is $\{s^k_{0,i}\}_{i\in N_0, k\in\mathcal{T}_i}$.

Since travel times are rational, there exists a $\eta > 0$ such that $\tau_i/\eta \in \mathbb{Z}$, for all $i \in N$. This η guarantees that starting at any time t^k_i, $k \in \mathcal{T}_i$, the arriving times to other locations $j \neq i$, are at times $t^k_i + p\eta$, with $p \in \mathbb{Z}_+$. At a given iteration, the algorithm 3 can correct the travel time starting from an initial time point, $s^k_{0,i}$; or from a time point added by algorithm 3, $s^k_{1,i}$; or from a time point added by algorithm 4, $s^k_{2,i}$; $i \in N$, $k \in \mathcal{T}_i$; or from a time point added by algorithm 5, $s^k_{3,i}$; $i \in N$, $k \in \mathcal{T}_i$. For each time $s^k_{0,i}$, $s^k_{1,i}$, $s^k_{2,i}$ and $s^k_{3,i}$ we consider all possible new times can be added starting from that time, i.e., $|N_0|\lceil H/\eta\rceil$ points. However, note that times $s^k_{1,i}$ don't need to be counted, since they are already considered for some time point t^ℓ_j, $j \in N_0$, $\ell \in \mathcal{T}_j$, from where $s^k_{1,i}$ was generated. Using lemma 5 at most $|N|\lceil 2H/\epsilon\rceil$ time points are added by algorithm 4 and by lemma 6 at most $2|N|\lceil 2H/\epsilon\rceil$, so no more than $\left(3|N|\lceil 2H/\epsilon\rceil + |\{s^k_i\}_{i\in N_0,k\in\mathcal{T}_i}|\right)\times|N_0|\lceil H/\eta\rceil$ are added by algorithm 6 for a given ϵ. □

Theorem 4. *Consider a feasible CIRP-OB instance. There exists an $\epsilon > 0$ such that Algorithm 6 finds an optimal solution.*

Proof. For any feasible instance of CIRP-OB there exists a rational solution, then there must be a $\epsilon_1 > 0$ such that for any two consecutive visits to a customer $i \in N$, the time between those visits is at least ϵ_1. In addition, by theorem 3, there exists a

$\epsilon_2 > 0$ such that the optimal solution for the LBM doesn't change after restoring the customer storage capacity, i.e., changing the upper bound for inventory levels from $C_i + \bar{u}_i \leq C_i + u_i\epsilon_2$ to C_i, $i \in N$. Consider $\epsilon = \min\{\epsilon_1, \epsilon_2\}$.

By proposition 35 we know the algorithm 6 finishes after a finite number of steps with ϵ. If an optimal solution is found before the last iteration the algorithm stops. If not, then a solution whose cost is a lower bound to the problem is provided by the LBM. This solution has no multiple visits at the same time since the algorithm 5 includes constraints at each time there are more than one visit. The algorithm maintains the valid lower bound condition of the LBM, since $\epsilon \leq \epsilon_1$. In addition, for this solution, the travel times are all corrected and the customers storage capacity is at most $C_i + u_i\epsilon$, $i \in N$. Since $\epsilon \leq \epsilon_2$, the inventory levels can be changed to C_i and the optimal solution is the same. Thus, the three relaxations for the LBM listed in section 4.5 are corrected, so the LBM is feasible and also is an optimal solution for the CIRP-OB. □

In practice $\epsilon > 0$ needed for Algorithm 5 might be too small and solving the LBM with a time-expanded network whose intervals are less than ϵ can be intractable. In the implementation the time-expanded network is refined using algorithms 3 and 4 (steps 7 and 8) only, and in case no points are added for those algorithms, it stops. Note that in this variant, no matter how large or small the input ϵ is, the LBM is always a lower bound model for the problem. This property allows us to use any discretization. We have seen that solving the LBM in course time discretizations leads to solution that are in many cases optimal. In addition, the model solves faster. We start the algorithm with a ϵ tolerance and in case no more refinements are made to the network, is reduced to $\epsilon/2$. This allows us to adjust the number of time points added by algorithm 4 as needed.

As the time expanded network is refined, the LBM finds better (less travel times or capacity violations) integer solutions when running the branch-and-bound. We can

take advantage of this exploration by running the FCM for each new integer solution is found, so it is possible to get feasible solutions before the algorithm finishes.

4.8 Computational Experiments

4.8.1 Instances

For our computational experiments, there are parameters and conditions that are the same for all generated instances. These are the following:

- vehicle capacity $Q = 50$;
- time horizon $H = 20$;
- initial inventory I_i^0 whose value is equal to customer storage capacity C_i for all $i \in N$;
- speed is equal to one;
- travel times are symmetric and equal to transportation costs.

The parameters that vary among the instances are determined in such a way that we study the impact of four factors on the performance of the algorithm:

- number of customers;
- geographical distribution of locations;
- percentage of customers whose storage capacity is less than the vehicle capacity Q, N_ℓ;
- customer usage rates.

The number of customers is a natural dimension to consider in our instances. We consider instances with up to 30 customers and three different values: 10, 20 and 30.

The customer locations with respect to the depot is a factor that impacts on the algorithm performance. Note that in the CIRP-OB problem, only the distance to the depot is relevant, so we consider that the customers are located in a line in which the depot is at position 0. Customer locations are randomly generated with an average distance from the depot of 5. The sample is given by a uniform Unif$(5-\delta, 5+\delta)$ distribution, and we consider three different values for $\delta = 0.5, 1.5, 2.5$.

Another factor that is important to include is the number of customers whose storage capacity is less than the vehicle capacity, since for these customers, vehicle waiting is allowed. We generate instances with 25%, 50% and 75% of customers with capacity $C_i < Q$. For each customer in N_ℓ, the value of C_i is an integer uniformly distributed in the interval $\left[\frac{Q}{2}, Q\right)$ and for each customer in N_g, C_i is an integer uniformly distributed in $\left[Q, \frac{3Q}{2}\right)$.

Once the geographical distribution and the storage capacities are sampled, we generate customer usage rates. For each $i \in N$, we define $low_i = \frac{C_i}{H}$ as the lower limit for u_i; a lower value for u_i implies the customer needs no visits from the depot. We also define an upper value for u_i, $up_i = \frac{\min\{C_i, Q\}}{2\tau_i}$. We consider three usage levels for instance generation, each level determine the range from which we sample usage values. For instances with a low usage rate (level 1) we get usage rates from Unif $\left(low_i, \frac{low_i + up_i}{2}\right)$; for instances with medium rate (level 2) we get rates from Unif(low_i, up_i); and for high rate (level 3) we get rates from Unif $\left(\frac{low_i + up_i}{2}, up_i\right)$, for all $i \in N$.

We generate 81 different instance types, each one is labeled N for the number of customers, G for the geographical distribution, P for the percentage of customers in N_ℓ, and U for the usage level. For example, an instance labeled N20G2P25U3 has 20 customers, the customer location distributions are sampled using Unif $(3.5, 6.5)$, it has 5 customers with storage capacity $C_i < Q$, and usage rates are sampled using Unif $\left(\frac{low_i + up_i}{2}, up_i\right)$ for all customers $i \in N$.

For each instance type, we generate three samples. In order to find the number of vehicles m for each sample, we run the UBM (described in section 4.6.1) with objective function minimizing the number of vehicles. The time discretization for the UBM is homogeneous, $\Delta = t_i^k - t_i^{k-1}$, $i \in N$, $k \in \mathcal{T}_i$, with $\Delta > 0$. Let $\Delta' > 0$ be the maximum value that satisfies conditions (4.19) and (4.20). We consider $\Delta = \min\{\Delta', 0.25\}$ for the UBM. The model runs for up to two hours, and the number of vehicles m we set for the instance is the best value found. Then, we run the DDD algorithm. We have seen that randomly generated instances likely result in instances that are solved in one iteration of the DDD algorithm. We have identified these instances before running our experiments and replaced them with re-sampled instances that require more iterations. We repeat the procedure until we have 3 samples for each of the 81 instance types. The generated instances can be found at `https://github.com/felipelagos/cirplib`.

4.8.2 Results

We generate three samples for each of the 81 types of instances, a total of 243 instances. We run the DDD algorithm for each instance for up to 2 hours. The algorithm starts with $\epsilon = 1$, which is reduced by half ($\epsilon \leftarrow \epsilon/2$) if no new time points are added to the time-expanded network. The algorithm either finishes with a provable optimal solution, with a feasible solution (and an optimality gap), or with no solution. As mentioned in Section 4.7, the routine for correcting the number of visits is not included in the algorithm. However we have seen, in our experiments, that no multiple vehicles visit a customer at the same time in the final solution for the instances with feasible solutions (no optimality proved in two hrs). In practice, for randomly generated instances, multiple visits at the same time relaxation does not have to be addressed.

Table 4.1 summarizes the results by different instance dimensions: number of

Table 4.1: Results summary.

	N			U			G			P		
	10	**20**	**30**	**1**	**2**	**3**	**1**	**2**	**3**	**25**	**50**	**75**
Optimal	80	69	57	80	72	54	79	68	59	68	72	66
Feasible	1	10	14	0	8	17	2	9	14	5	8	12
No Solution	0	2	10	1	1	10	0	4	8	8	1	3
Vehicles	7.83	15.47	23.64	13.36	15.67	17.91	15.99	15.65	15.30	15.15	15.67	16.12
Visits	13.52	25.86	37.92	21.85	25.61	28.63	25.85	24.90	24.92	22.79	25.41	27.35
Time Points	23.22	36.07	42.09	34.43	33.16	33.79	29.20	33.61	38.56	37.42	32.61	31.35
Initial Points (%)	27.45	17.54	13.60	17.16	19.59	21.84	21.67	19.47	17.45	18.99	20.16	19.43

customers (N); usage rate level (U); geographical distribution (G); and percentage of customers in N_ℓ (P). The table shows the number of instances for which the algorithm finishes with an optimal solution (Optimal), with a feasible solution (Feasible), or with no solution (No Solution). We also show the average number of vehicles (Vehicles) of the instance and the average number of visits (Visits) of the found feasible solution. Finally, we report the average number of time points added to a customer time discretization (Time Points) and the percentage that represents the initial number of time points (starting time-expanded network) with respect to the total number at the last iteration (Initial Points).

Out of the 243 instances, the DDD algorithm optimally solves 206 of them (84.77%), finds a feasible solution (with a positive gap) for 25 instances (10.29%), and finds no solution for 12 instances (4.94%) (Recall that these instances were selected to be "difficult" for the DDD algorithm). In the detail for the number of customers, almost all of the instances are optimally solved for N10, but only 57 instances out of 81 are optimally solved for N30. In general, we see that as we increase the number of customers, the problem becomes more difficult to solve. We see a similar trend for the usage rate: as the level increases, fewer instances are solved optimally, and more instances have an optimality gap or no solution. When the geographical customer distributions are analyzed, the clustered instances (G1: $\delta = 0.5$, no more than 0.5 units of distance to the depot) are easier to solve than the non-clustered instances. The percentage of customers in N_ℓ does not show a clear impact on the algorithm.

In Table 4.1, we also see that the number of vehicles needed for the instance is

related to the number of customers and the usage rate level. In both cases, as the number increases (N10 to N30, U1 to U3), more vehicles are necessary. However, the customer distribution (G) and the percentage of N_ℓ (P) have no clear impact on the number of vehicles.

The number of visits in the feasible solution depends on N, U, and P, but not on G. Note that if the instance has more customers in N_ℓ, then more visits have to be made, independent of the usage rate level: instead of waiting, the vehicles make more out-and-back trips to customers whose capacity C_i is less than Q.

For N10 instances, each customer has, on average, about 23 different time points in its time discretization set when the algorithm stops. When the number of customers is N20, the average is about 36, and for N30, about 42. This indicates that the more customers that are in the instance, the more time points that are needed, possibly because there is more flexibility in the vehicle itineraries (more options for how to combine customer visits in a vehicle itinerary). Another dimension that affects the number of visits is the geographical distribution: customers in clustered instances (G1) need fewer time points than customers in non-clustered instances (G3). Note that for clustered instances, the travel times τ_i are similar for all $i \in N$. Thus, the vehicle visit customers at similar times, which is not true for non-clustered instances.

Figure 4.4 reports the experiments running time. For each dimension (N, U, G, P), a plot with the percentage of instances solved for a given time, from 0 to 7200 seconds (2 hours), is set. The number of customers plot shows curves that are more separated each other, confirming that this dimension is critical in order to explain the difficulty of an instance. Most of the N10 instances are optimally solved in less than 20 minutes; about a 80% of the N20 instances are solved in less than one hour, while 70% of the N30 instances require two hours to be optimally solved. A similar trend is seen for usage and geographical dimensions. Instances with levels U1 and U2 require less solving time than U3 instances, while clustered instances are the fastest to solve

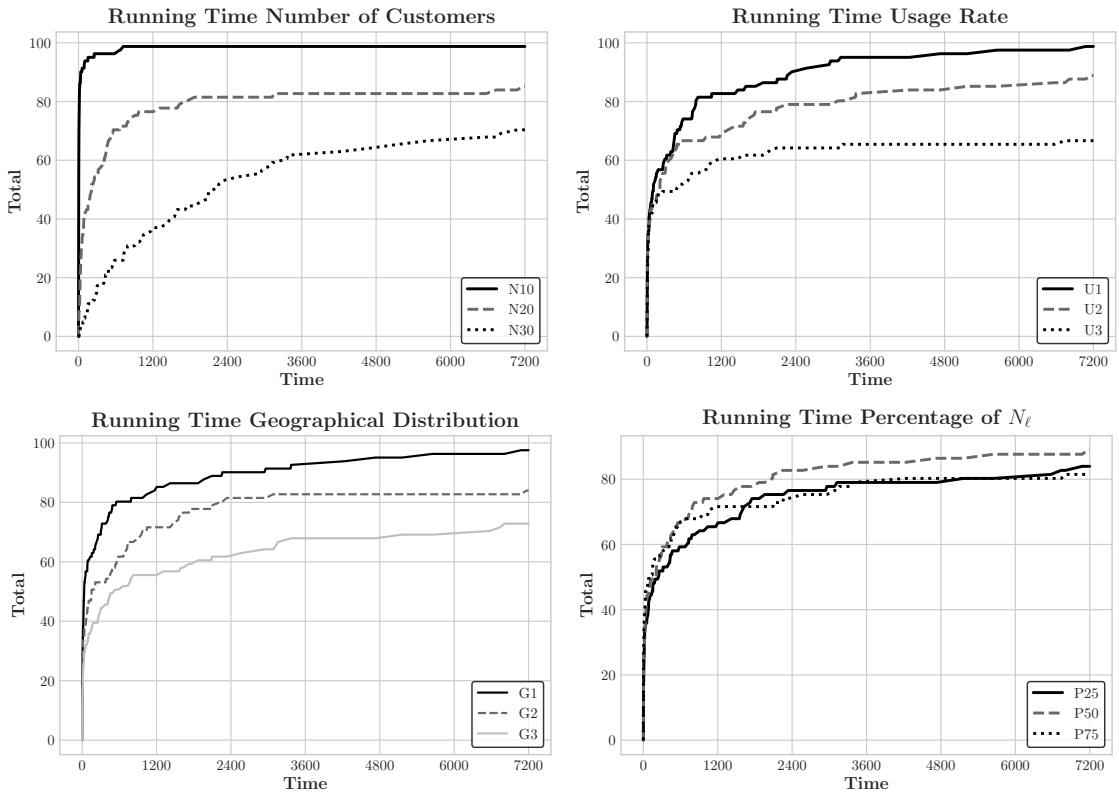

Figure 4.4: Running Time

for G. When the number of customers N_ℓ varies, the running time of the instance is not affected.

In Figure 4.5, we plot the optimality gap of the solution found by the DDD algorithm in two hours. The curve corresponds to the percentage of instances whose gap is less than a given value. For the number of customers (N) plot, the only instance with 10 customers, N10, the algorithm doesn't optimally solve in two hours has a small gap. For N20 instances, about 95% have a gap of less than 5%, and for N30 instances, about 85% have a gap of less than 10%. For the usage rate (U), almost all of the U1 and U2 instances have a gap of less than 8%. The plot also shows that, in general, the U3 instances have larger optimality gaps.

In Figure 4.6, we show the percentage of instances for which the DDD algorithm finishes in less than or equal to a given number of iterations (independent of the status of the solution). We see that a significant percentage of the instances requires

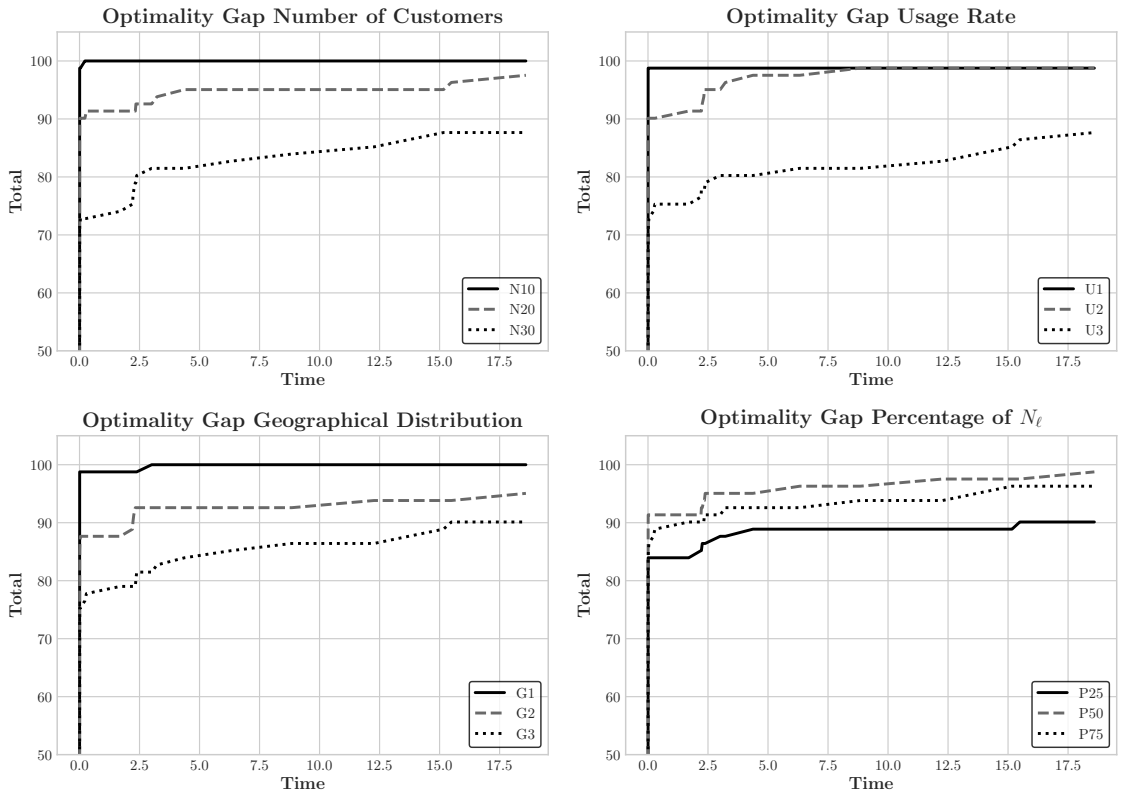

Figure 4.5: Optimality Gap

more than 50 iterations. In general, the conclusions obtained from previous plots and tables are confirmed when analyzing the number of iterations.

Table 4.2: Status for DDD algorithm without strengthenings.

	All	**No Branch**	**No Valid**	**No Visits**	**No Wait**
Optimal	69	69	10	68	66
Feasible	10	9	69	9	11
No Solution	2	3	2	4	4

Table 4.2, which presents, in part, the impact of different strengthenings described in Section 4.5, shows five different variants of the DDD algorithm:

- **All**: complete DDD algorithm;
- **No branch**: the branch-and-bound strategy in section 4.5.4 is not used;
- **No Valid**: the valid inequalities in section 4.5.3 are not included;

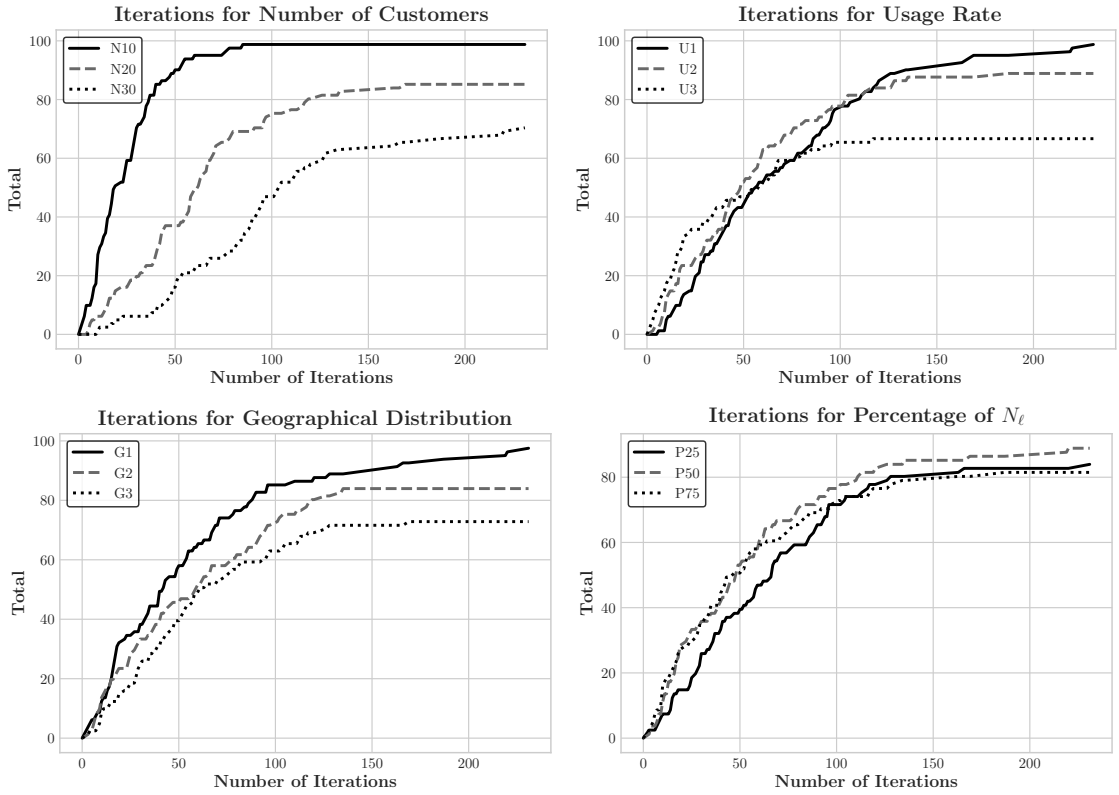

Figure 4.6: Number of iterations

- **No Visits**: the conditions in section 4.5.2 for visits times in the LBM are not used;

- **No Wait**: the conditions in section 4.5.1 for vehicles waiting in the LBM are not included.

For each variant, the number of instances with an optimal, a feasible or no solution status are computed for N20 instances (20 customers) after two hours running. All strengthenings lead to improvement in the algorithm's performance: for all the cases, the full DDD algorithm (All) has more instances with a feasible or an optimal solution when some variant does not. However, when the valid inequalities are turned off, the impact is more significant on the number of instances with optimal solutions than it is in any other case (not in the number of 'No Solution' status).

In Figure 4.7, we consider the same DDD algorithm variants presented in Table

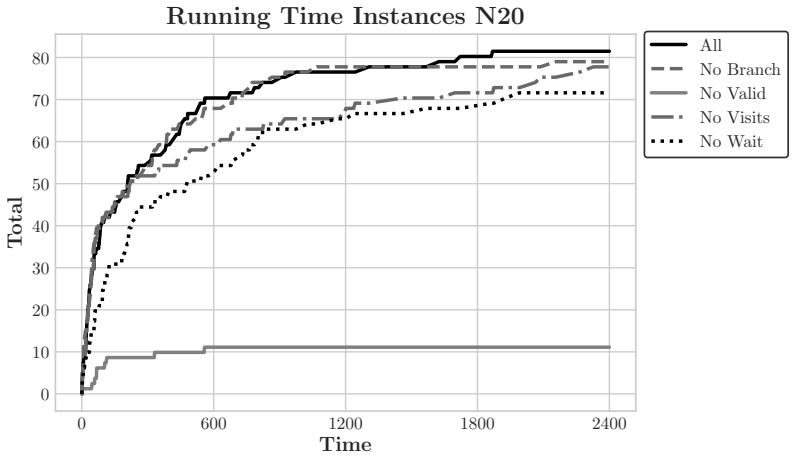

Figure 4.7: Running time for the DDD algorithm without strengthenings.

4.2 in a running time plot, which shows the percentage of instances solved optimally in a given time. The plot considers the first 20 minutes and validates the previous observation: the valid inequalities impact the performance the most. We also see that the waiting-vehicles and the visit times conditions (Section 4.5.1 and 4.5.2) have a significant effect on the algorithm running times, especially during the first 10 minutes. Figure 4.7 also indicates that when the branch-and-bound strategy is not included, the impact is not so relevant. One explanation for this is that the improvement of this strategy is already implied by other strengthening components.

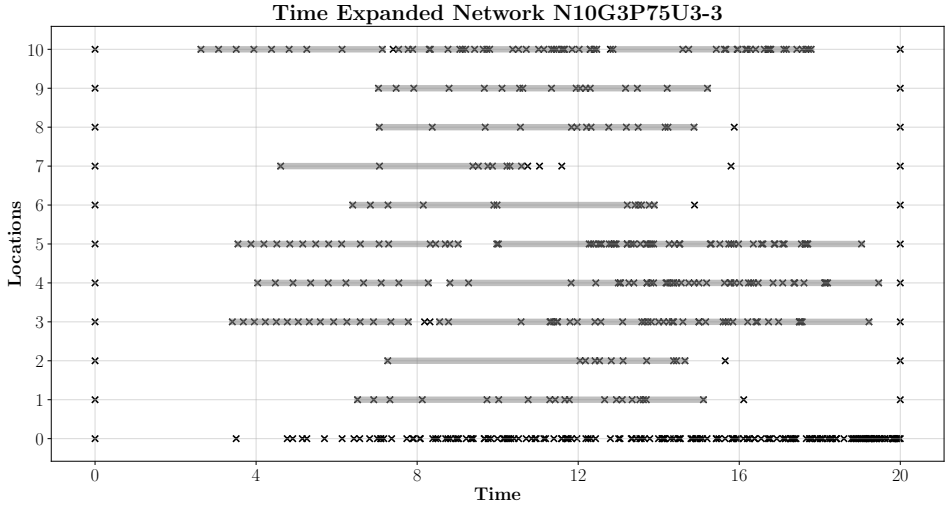

Figure 4.8: DDD algorithm time discretization example.

As an example of the DDD algorithm, we plot the time expanded network for the instance N10G3P75U3-3 (the third sample of type N10G3P75U3) in Figure 4.8. This instance is solved optimally by the algorithm, and the network is the time discretization corresponding to the last iteration (54 iterations). The crosses represent the time points for each location (0 corresponds to the depot), and the gray lines indicate the time intervals in equations (4.14) and (4.15). These intervals are the visit arrival times when the number of visits is the minimum to deliver the customer usage during the time H. We first note that most of the time points are in the intervals, which shows that these intervals are relevant and that the algorithm does not need to fully discretize time in order to solve the problem. For Customer 9, for example, the time interval that is relevant for the (only) visit to the customer is from time 7 to 15. Customer 5 has two visits, one between time 3.5 and 9, another between 10 and 19.

The DDD algorithm efficiently and dynamically discretizes the time-expanded network, focusing only on relevant time intervals.

4.9 Discussion and Future Research

In this work, we study a special case of the CIRP, the CIRP-OB, and we propose a new and efficient algorithm for solving it. We successfully implement the DDD algorithm, an algorithm that has been used to solve several continuous time problems, and our computational experiments validate the potential of this algorithm. Out of 243 randomly generated CIRP-OB instances, the DDD algorithm finds a provable optimal solution for 84% of them in less than two hours, and for 95%, the algorithm returns a feasible solution with an optimality gap of less than 18%.

To the best of our knowledge, this is the first time a DDD algorithm has been developed for a Continuous Time Inventory Routing Problem. [72], recently studied the CIRP and developed an algorithm that relies on an IP formulation over a

time-expanded network, and can be viewed as a partial implementation of a DDD algorithm. Their methodology used models constructed over homogeneous time discretizations. In this work, we study complex models and algorithms in non-homogeneous time expanded networks and show the power of using strategies that dynamically discretize time.

However, the DDD algorithm implementation is not straightforward for the CIRP-OB. The LBM formulation presents some challenges that need to be addressed. One is that a very fine time discretization does not lead to an exact formulation for the continuous time problem. In this formulation, the customer storage capacities are always strictly greater than the real capacity C_i, $i \in N$. For other problems in which the DDD algorithm has been studied (e.g. [8]), this does not happen, and the lower bound model eventually becomes an exact model for the problem. Nevertheless, we overcome this problem by using Theorem 3, which allows us to prove that the DDD algorithm is an exact algorithm for the problem in that the DDD algorithm always returns an optimal solution for a feasible CIRP-OB instance.

In our experiments, we study several instance dimensions: the number of customers (N), the geographical distribution of the locations (G), the percentage of customers in N_ℓ (P), and the usage rate level (U). Of these, variations on the number of customers and the usage rate better explain the complexity of the problem (running time, optimality gap, iterations). We also see that the percentage of customers in N_ℓ (P) does not have a clear impact on the instance difficulty. The DDD algorithm can optimally solve instances with up to 30 customers, random geographical locations, and high usage rates. The algorithm efficiency results, in part, from the fact that the time discretization is more intensive in particular time intervals, so not all of time horizon H is considered.

In this work, we show the potential of using a dynamic discretization algorithm for routing problems. Some ideas presented in this work can be extended to the CIRP;

this is the first step in solving the CIRP. One problem that arises when solving IRP problems is that exact time indexed formulations are weak (as shown in Section 4.3), so the structure of the problem solution needs to be exploited. For future research, several conditions and valid inequalities we propose for the LBM model can be used for the CIRP.

Appendices

APPENDIX A

BRANCH-AND-PRICE FOR ROUTING WITH PROBABILISTIC CUSTOMERS

We present three tables with additional results. Tables A.1 and A.2 are analogues of Table 2.3 for $p = 0.5$ and $p = 0.9$, respectively. Table A.3 displays average running times for B&P algorithms with UCA and FCA.

Table A.1: Average relative gap of solution expected cost for UCA, FCA and deterministic problem, for realization probability 0.5.

Customers	Steps (K)	Type C			Type R		
		UCA Exact / Best LB	FCA / Best LB	Deter / Best LB	UCA Exact / Best LB	FCA / Best LB	Deter / Best LB
15	0	2.75%	2.67%	2.91 %	1.79%	2.04%	2.21 %
	1	1.43%	2.28%		1.24%	1.95%	
	2	1.01%	1.15%		1.23%	1.19%	
	3	0.92%	0.89%		1.14%	1.14%	
	4	0.78%	0.82%		1.28%	1.12%	
25	0	4.15%	4.19%	5.32 %	3.17%	3.51%	3.54 %
	1	4.44%	4.99%		2.66%	3.34%	
	2	3.83%	4.47%		2.45%	2.78%	
	3	3.1%	3.29%		1.6%	1.7%	
	4	1.35%	1.42%		1.65%	1.57%	
40	0	6.42%	6.42%	7.43 %	4.49%	4.52%	4.66 %
	1	7.43%	7.43%		4.22%	4.52%	
	2	7.25%	7.43%		3.76%	4.14%	
	3	7.25%	7.24%		2.29%	2.4%	
	4	5.29%	5.53%		1.79%	1.43%	

Table A.2: Average relative gap of solution expected cost for UCA, FCA and deterministic problem, for realization probability 0.9.

Customers	Steps (K)	Type C			Type R		
		$\frac{\text{UCA Exact}}{\text{Best LB}}$	$\frac{\text{FCA}}{\text{Best LB}}$	$\frac{\text{Deter}}{\text{Best LB}}$	$\frac{\text{UCA Exact}}{\text{Best LB}}$	$\frac{\text{FCA}}{\text{Best LB}}$	$\frac{\text{Deter}}{\text{Best LB}}$
15	0	0.52%	0.53%	0.55 %	0.11%	0.11%	0.11 %
	1	0.31%	0.31%		0.04%	0.04%	
	2	0.31%	0.31%		0.04%	0.04%	
	3	0.31%	0.31%		0.04%	0.04%	
	4	0.31%	0.31%		0.04%	0.04%	
25	0	0.97%	0.98%	2.1 %	0.43%	0.56%	0.58 %
	1	2.1%	2.1%		0.51%	0.51%	
	2	2.1%	2.1%		0.39%	0.39%	
	3	1.67%	1.77%		0.11%	0.11%	
	4	0.56%	0.56%		0.11%	0.11%	
40	0	2.46%	2.46%	3.51 %	0.77%	0.77%	0.89 %
	1	3.13%	3.13%		0.77%	0.87%	
	2	3.13%	3.13%		0.45%	0.45%	
	3	1.76%	1.76%		0.49%	0.49%	
	4	2.29%	3.13%		0%	0%	

Table A.3: Average running time for UCA and FCA in seconds.

Customers	Steps (K)	Type C		Type R	
		UCA	FCA	UCA	FCA
15	0	729	361	99	717
	1	1552	878	838	1273
	2	2842	2018	2021	1636
	3	2803	2992	2488	1850
	4	4545	4054	2968	1922
25	0	6909	4736	1497	2205
	1	10622	8822	2945	2951
	2	12714	11462	3700	3571
	3	13715	13803	4989	5021
	4	13794	13431	6691	5983
40	0	12474	8623	8124	3630
	1	19704	16911	8282	7489
	2	20190	20638	9374	9273
	3	20669	21154	10214	9619
	4	21600	20700	10490	9738

APPENDIX B

THE CONTINUOUS TIME INVENTORY ROUTING PROBLEM

Appendix 1 – Relation to the model of [2]

In the model presented in [2], the customer inventory levels are managed by introducing lower and upper bounds on the (cumulative) delivery quantity based on customer usage rates. This is equivalent to managing inventory levels imposing inventory flow conservation. Indeed, in [2], the following constraints ensure that the total delivery to customer i during the time interval from 0 to t is feasible,

$$d_{it} \leq \sum_{k \in R_i} \sum_{v \in V_k} \sum_{\tau \in T_k : \tau \leq t - tc_{ik}} x_{ik\tau v} \leq D_{it}, \qquad \text{for all } t \text{ and } i \in A_1,$$

where d_{it} is the lower limit on the amount that must be delivered to customer i during the interval from 0 to t, D_{it} is the upper limit on the amount that can be delivered to customer i during the interval from 0 to t, and $x_{ik\tau v}$ is a variable that represents the amount delivered to customer i on route k by vehicle v starting at time τ (and R_i the set of routes that visit i, V_k the set of vehicles that can driver route k, T_k the set of time points at which route k can start, tc_{ik} the time after the start of route k until delivery at customer i is completed, and A_1 the set of customers that can receive any number of deliveries). Note that $\sum_{k \in R_i} \sum_{v \in V_k} \sum_{\tau \in T_k, \tau \leq t - tc_{ik}} x_{ik\tau v}$ represents the total delivery to customer i up to time t.

In our setting, the total delivery to customer i from 0 to t is the following,

$$\sum_{s \leq t} y_i^s = \sum_{0 < s \leq t} (z_i^s - z_i^{s-1} + u_i) + (z_i^0 - I_i^0) = z_i^t + u_i(t-1) - I_i^0.$$

Using the inventory bounds, $u_i \leq z_i^t \leq C_i$, we have

$$u_i t - I_i^0 \leq \sum_{s \leq t} y_i^s \leq C_i - I_i^0 + u_i(t-1),$$

where $u_i t - I_i^0$ is the lower limit on the amount that must be delivered to i (i.e., d_{it}), and $C_i - I_i^0 + u_i(t-1)$ is the upper limit (i.e., D_{it}).

Thus, the model presented in [2] has "substituted out" the inventory balance constraints.

Appendix 2 – Calculating big-M values for depot subtour elimination

Here we consider some possible ways to determine the LBM parameter, M_{ij}^t.

If all (original) travel times between pairs of customers are positive, say $\min_{i,j \in N_0, i \neq j} \hat{\tau}_{ij} =: \epsilon > 0$, then the number of vehicle arrivals from another customer (or the depot) in the time remaining after the arrival at j (following a departure from i at time $t\Delta$), may be at most $(H - (t\Delta + \hat{\tau}_{ij}))/\epsilon$. Also, a vehicle can arrive by waiting, of which there can be at most $T - (t + \tau_{ij})$ cases. However, if $\Delta < \epsilon$, then depot subtours cannot occur, and otherwise, the upper bound obtained by "using" an interval of length Δ to move between customers taking time ϵ is greater than 1, which is all that can be "used" by waiting, so $(H - (t\Delta + \hat{\tau}_{ij}))/\epsilon$ is a valid upper bound on the number of arrivals. Thus we may take $M_{ij}^t = 1 + \lfloor (H - (t\Delta + \hat{\tau}_{ij}))/\epsilon \rfloor$. (The extra 1 is to count the arrival at j after departure from i at time point t.)

Otherwise, if there are co-located customers, for example, so $\hat{\tau}_{ij} = 0$ for some pair $i \neq j$, then if all original parameters are integer, we may use the observation that the exact MIP model has integer solutions, to see that all customer visits must deliver at least 1 unit of product. So we may take $M_{ij}^t = 1 + \sum_{k \in N} \hat{u}_k (H - (t\Delta + \hat{\tau}_{ij})$, which is an upper bound on the number of units of product that can be delivered in the time remaining.

Appendix 3 – Route-preserving only (RPO) model

$$\max \quad \zeta$$

$$\text{s.t.} \quad \bar{t}_r^k + \tau_{\rho(r,k)\rho(r,k+1)} \le \bar{t}_r^{k+1} \qquad r \in R,\ k = 0, \ldots, n_r$$

$$\bar{t}_r^{n_r+1} \le \bar{t}_{\bar{r}}^0 + H(1 - y_{r\bar{r}}) \qquad r \in R, \bar{r} \in R, r \ne \bar{r},$$

$$\sum_{\bar{r} \in R} y_{r\bar{r}} \le 1 \qquad r \in R,$$

$$\sum_{\bar{r} \in R} y_{\bar{r}r} \le 1 \qquad r \in R,$$

$$\sum_{r,\bar{r} \in R} y_{r\bar{r}} \ge |R| - m,$$

$$\sum_{k=1}^{n_r} \bar{q}_r^k \le Q \qquad r \in R$$

$$t_i^\ell \ge t_i^{\ell-1} + \zeta \qquad i \in N_0, \ell = 1, \ldots, n_i$$

$$-H(1 - x_{rk}^{\rho(r,k)\ell}) \le t_{\rho(r,k)}^\ell - \bar{t}_r^k \le H(1 - x_{rk}^{\rho(r,k)\ell}) \qquad r \in R,\ k = 1, \ldots, n_r,\ \ell = 1, \ldots, n_{\rho(r,k)},$$

$$-Q(1 - x_{rk}^{\rho(r,k)\ell}) \le q_{\rho(r,k)}^\ell - \bar{q}_r^k \le Q(1 - x_{rk}^{\rho(r,k)\ell}) \qquad r \in R,\ k = 1, \ldots, n_r,\ \ell = 1, \ldots, n_{\rho(r,k)}$$

$$\sum_{\ell=1}^{n_{\rho(r,k)}} x_{rk}^{\rho(r,k)\ell} = 1 \qquad r \in R,\ k = 1, \ldots, n_r$$

$$\sum_{r \in R} \sum_{k=1,\ldots,n_r\,:\,\rho(r,k)=i} x_{rk}^{\rho(r,k)\ell} = 1 \qquad i \in N,\ \ell = 1, \ldots, n_i$$

$$I_i^\ell = I_i^{\ell-1} - u_i \left(t_i^\ell - t_i^{\ell-1}\right) + q_i^\ell \qquad i \in N,\ \ell = 1, \ldots, n_i$$

$$I_0^i \ge u_i H \qquad i \in N \text{ and } n_i = 0$$

$$I_i^{n_i} - u_i(H - I_i^{n_i}) \ge 0 \qquad i \in N \text{ and } n_i > 0$$

$$q_i^\ell \le I_i^\ell \le C_i \qquad i \in N,\ \ell = 1, \ldots, n_i$$

$$q_i^\ell \ge 0 \qquad i \in N,\ \ell = 1, \ldots, n_i$$

$$t_i^\ell \ge 0 \qquad i \in N,\ \ell = 1, \ldots, n_i$$

Appendix 4 – Detailed computational results for a few instances

Note that the time limit is set to two hours of cpu time, using the equivalent in terms of "ticks" (see CPLEX manual for details). However, we report the observed wall clock time (which can be much more than the cpu time). Note too that UBM was only run for values of $H/\Delta \geq 9$. Finally, we indicate that no feasible solution was found with a dash (-); if no feasible solution to LBM is found, then it is not possible to run RPO. Recall that when the objective function value for RPO is positive, a feasible solution with value equal to the value of the solution to LBM has been constructed.

Instance	H/Δ	LBM solution time	LBM bound value	LBM optimality gap	RPO solution time	RPO solution value	UBM solution time	UBM solution value	UBM optimality gap
R5U2Q2	1	1.86	36.42	0.00	0.01	-			
	2	12.91	36.42	0.00	0.01	-			
	3	10.17	36.42	0.00	0.09	-			
	4	535.68	36.42	0.00	0.13	-			
	5	1997.23	36.42	0.00	0.07	-			
	6	8752.82	36.42	0.00	8.99	-			
	7	TL	33.68	8.13	0.45	-			
	8	TL	34.14	6.67	2.16	-			
	9	1751.35	36.42	0.00	0.05	-	10.20	39.23	0.00
	10	TL	35.68	2.09	1.72	-	154.56	39.23	0.00
	12	TL	34.81	4.63	2.68	-	6.93	-	-
	14	TL	34.54	5.46	0.16	-	15.86	-	-
	16	TL	32.91	10.67	14.82	-	48.40	39.32	0.00
	18	TL	31.56	15.41	19.13	-	70.37	36.59	0.00
	24	TL	32.46	12.21	8270.25	-	308.10	36.68	0.00
	30	TL	31.57	15.35	24.17	-	7312.31	36.59	0.00
	36	TL	31.51	15.88	0.01	0.43	17030.43	36.59	0.00
	42	TL	31.52	15.81	TL	-	TL	36.59	6.73
	48	TL	31.53	15.50	TL	-	TL	36.59	13.68
	54	TL	31.48	15.99	0.02	0.15	TL	36.59	8.59
	60	TL	31.48	15.99	127.35	0.29	TL	36.59	15.84

Instance	H/Δ	LBM solution time	LBM bound value	LBM optimality gap	RPO solution time	RPO solution value	UBM solution time	UBM solution value	UBM optimality gap
R10U1Q2	1	26.61	64.02	0.00	0.02	-			
	2	482.30	64.02	0.00	0.02	-			
	3	TL	62.80	1.94	6.55	0.73			
	4	TL	62.16	3.00	5.92	-			
	5	TL	59.81	7.04	0.04	-			
	6	TL	58.83	8.82	297.88	-			
	7	TL	57.54	11.26	0.84	0.76			
	8	TL	56.99	12.34	78.54	0.12			
	9	TL	57.51	11.33	37.16	-	305.50	64.16	0.00
	10	TL	57.11	12.10	11820.51	-	3157.92	64.30	0.00
	12	TL	56.88	12.55	651.73	-	TL	66.80	8.00
	14	TL	60.46	5.89	0.02	0.79	1913.11	65.94	0.00
	16	TL	59.18	8.18	0.02	0.84	TL	65.94	9.36
	18	TL	59.18	8.19	398.85	-	TL	64.11	5.60
	24	TL	58.75	8.98	3.63	0.50	TL	64.02	8.92
	30	TL	57.53	11.28	0.11	0.40	TL	64.16	9.69
	36	TL	57.62	11.10	0.06	0.50	TL	64.11	11.30
	42	TL	57.56	11.22	0.08	0.41	TL	65.85	14.97
	48	TL	56.75	-	-	-	TL	68.93	18.89
	54	TL	56.72	12.88	0.23	0.42	TL	-	-
	60	TL	56.72	-	-	-	TL	-	-

Instance	H/Δ	LBM solution time	LBM bound value	LBM optimality gap	RPO solution time	RPO solution value	UBM solution time	UBM solution value	UBM optimality gap
R10U2Q2	1	8.60	82.71	0.00	0.01	-			
	2	TL	82.28	0.52	0.00	-			
	3	TL	78.10	5.91	0.01	-			
	4	TL	74.43	11.12	51.30	-			
	5	TL	74.36	11.22	68.61	-			
	6	TL	72.87	13.50	93.59	-			
	7	TL	72.41	14.23	0.01	-			
	8	TL	72.21	14.54	1.62	-			
	9	TL	72.66	13.83	8.82	-	1133.12	-	-
	10	TL	72.94	13.40	629.83	-	TL	94.18	14.14
	12	TL	72.08	14.74	44.13	-	TL	-	-
	14	TL	73.57	12.43	3486.23	-	TL	98.56	12.66
	16	TL	73.35	12.77	TL	-	TL	92.65	18.23
	18	TL	73.62	12.35	TL	-	TL	87.92	11.32
	24	TL	73.45	12.61	TL	-	TL	-	-
	30	TL	72.22	14.53	TL	-	TL	-	-
	36	TL	72.46	-	-	-	TL	-	-
	42	TL	72.19	-	-	-	TL	-	-
	48	TL	72.08	-	-	-	TL	-	-
	54	TL	72.11	-	-	-	TL	-	-
	60	TL	72.55	18.31	0.07	0.30	TL	-	-

Instance	H/Δ	LBM solution time	LBM bound value	LBM optimality gap	RPO solution time	RPO solution value	UBM solution time	UBM solution value	UBM optimality gap
C15U2Q2	1	18.60	93.98	0.00	0.21	-			
	2	TL	90.42	3.93	0.02	-			
	3	TL	92.73	1.35	120.94	-			
	4	TL	86.58	8.55	2248.46	-			
	5	TL	86.47	8.68	80.82	-			
	6	TL	86.95	8.09	1998.54	-			
	7	TL	86.67	8.43	2403.24	-			
	8	TL	86.76	8.33	3383.00	-			
	9	TL	86.28	8.92	TL	-	5.62	-	-
	10	TL	86.94	8.10	TL	-	391.61	-	-
	12	TL	86.30	9.15	TL	-	3.22	-	-
	14	TL	86.27	8.94	TL	-	TL	-	-
	18	TL	86.33	9.28	TL	-	TL	-	-
	24	TL	86.29	9.78	TL	-	TL	97.49	6.21
	30	TL	86.27	10.36	TL	-	TL	97.00	12.44
	36	TL	86.27	-	-	-	TL	-	-
	42	TL	86.27	-	-	-	TL	-	-
	48	TL	86.27	-	-	-	TL	-	-
	54	TL	86.27	-	-	-	TL	-	-
	60	TL	86.27	-	-	-	TL	-	-

APPENDIX C
AN EXACT ALGORITHM FOR THE CONTINUOUS TIME INVENTORY ROUTING PROBLEM WITH OUT-AND-BACK ROUTES

Appendix 1 – Result Details

In Tables C.1, C.2 and C.3 the result details are shown. The Instance column has the instance name, corresponding to the type slash (-) sample number. The Lower column shows the best lower bound value found and the Best column, the best feasible solution value. The Runtime column indicates the running time of the DDD algorithm and the Iterations column, the total number of iterations. The Points column shows the total number of time points added to the time-expanded network (including all locations in N_0).

Table C.1: Instances with 10 customers (N10).

Instance	Lower	Best	Runtime	Iterations	Points
N10G1P25U1-1	110.08	110.08	4.79	33	220
N10G1P25U1-2	109.64	109.64	8.75	50	239
N10G1P25U1-3	110.98	110.98	3.64	30	183
N10G1P25U2-1	107.14	107.14	13.93	40	299
N10G1P25U2-2	109.36	109.36	7.20	35	252
N10G1P25U2-3	109.20	109.20	2.73	15	178
N10G1P25U3-1	107.20	107.20	0.90	3	98
N10G1P25U3-2	110.62	110.62	1.07	3	88
N10G1P25U3-3	110.92	110.92	2.27	10	154
N10G1P50U1-1	110.06	110.06	3.11	19	194
N10G1P50U1-2	110.56	110.56	14.01	55	296
N10G1P50U1-3	106.02	106.02	4.28	18	219
N10G1P50U2-1	108.62	108.62	1.79	8	133
N10G1P50U2-2	109.38	109.38	3.58	17	193
N10G1P50U2-3	109.44	109.44	2.80	17	174
N10G1P50U3-1	109.04	109.04	3.53	15	197
N10G1P50U3-2	107.30	107.30	2.77	13	171
N10G1P50U3-3	113.32	113.32	1.13	4	111
N10G1P75U1-1	109.02	109.02	12.61	53	311
N10G1P75U1-2	110.62	110.62	2.69	18	166
N10G1P75U1-3	108.74	108.74	4.82	35	224
N10G1P75U2-1	109.76	109.76	0.85	10	141
N10G1P75U2-2	98.90	98.90	1.92	18	174
N10G1P75U2-3	110.42	110.42	6.62	40	216
N10G1P75U3-1	106.90	106.90	2.07	16	180
N10G1P75U3-2	105.10	105.10	8.63	40	236
N10G1P75U3-3	105.02	105.02	0.35	3	90
N10G2P25U1-1	113.34	113.34	1.04	15	163
N10G2P25U1-2	112.04	112.04	3.21	25	238
N10G2P25U1-3	115.50	115.50	3.83	30	219
N10G2P25U2-1	110.38	110.38	2.33	24	183
N10G2P25U2-2	108.64	108.64	0.70	7	153
N10G2P25U2-3	108.96	108.96	20.45	37	314
N10G2P25U3-1	112.60	112.60	15.31	36	271
N10G2P25U3-2	123.18	123.18	721.17	85	565
N10G2P25U3-3	116.26	116.26	3.50	18	202
N10G2P50U1-1	100.98	100.98	1.46	13	190
N10G2P50U1-2	100.66	100.66	34.80	78	355
N10G2P50U1-3	111.62	111.62	0.78	11	143
N10G2P50U2-1	101.20	101.20	1.70	17	158
N10G2P50U2-2	110.08	110.08	1.65	10	188
N10G2P50U2-3	118.70	118.70	3.13	25	183
N10G2P50U3-1	88.76	88.76	0.96	8	153
N10G2P50U3-2	115.38	115.38	0.41	4	102
N10G2P50U3-3	106.68	106.68	0.72	7	140
N10G2P75U1-1	104.78	104.78	3.89	28	199
N10G2P75U1-2	113.52	113.52	0.66	10	143
N10G2P75U1-3	103.46	103.46	3.15	24	218
N10G2P75U2-1	108.10	108.10	0.85	12	129
N10G2P75U2-2	109.64	109.64	4.13	29	216
N10G2P75U2-3	108.72	108.72	1.04	10	161
N10G2P75U3-1	119.64	119.64	91.83	60	365
N10G2P75U3-2	122.54	122.54	33.94	34	294
N10G2P75U3-3	109.78	109.78	0.32	3	81
N10G3P25U1-1	107.38	107.38	10.77	37	278
N10G3P25U1-2	111.18	111.18	3.72	28	210
N10G3P25U1-3	102.28	102.28	3.78	25	248
N10G3P25U2-1	116.80	116.80	1.07	8	157
N10G3P25U2-2	99.68	99.68	16.96	47	284
N10G3P25U2-3	101.58	101.58	252.07	76	428
N10G3P25U3-1	119.72	119.72	11.14	15	304
N10G3P25U3-2	114.14	114.14	10.75	29	261
N10G3P25U3-3	118.28	118.28	1.63	9	184
N10G3P50U1-1	119.30	119.30	0.89	10	142
N10G3P50U1-2	108.22	108.22	3.15	24	180
N10G3P50U1-3	84.80	84.80	5.74	28	235
N10G3P50U2-1	121.62	121.62	58.05	30	314
N10G3P50U2-2	122.82	122.82	0.36	3	82
N10G3P50U2-3	112.10	112.10	1.60	10	181
N10G3P50U3-1	120.16	120.16	102.02	48	366
N10G3P50U3-2	123.70	123.70	12.31	22	247
N10G3P50U3-3	127.44	127.44	31.01	35	291
N10G3P75U1-1	115.14	115.14	1.18	14	175
N10G3P75U1-2	100.22	100.22	1.12	10	171
N10G3P75U1-3	98.08	98.08	14.76	43	294
N10G3P75U2-1	118.96	118.96	1.30	11	166
N10G3P75U2-2	123.12	123.12	6.24	31	199
N10G3P75U2-3	108.34	108.34	0.35	4	78
N10G3P75U3-1	120.20	120.20	152.49	29	396
N10G3P75U3-2	124.54	124.82	7200.00	88	749
N10G3P75U3-3	146.80	146.80	663.58	54	516

Table C.2: Instances with 20 customers (N20).

Instance	Lower	Best	Runtime	Iterations	Points
N20G1P25U1-1	214.48	214.48	317.91	163	757
N20G1P25U1-2	213.28	213.28	21.72	27	528
N20G1P25U1-3	208.16	208.16	90.14	71	701
N20G1P25U2-1	210.92	210.92	188.32	58	795
N20G1P25U2-2	211.22	211.22	35.95	30	535
N20G1P25U2-3	213.18	213.18	157.25	67	653
N20G1P25U3-1	208.92	208.92	19.59	16	313
N20G1P25U3-2	210.00	210.00	251.76	67	701
N20G1P25U3-3	215.44	215.44	20.07	14	370
N20G1P50U1-1	207.84	207.84	55.25	45	580
N20G1P50U1-2	208.56	208.56	130.29	69	721
N20G1P50U1-3	209.42	209.42	257.36	110	770
N20G1P50U2-1	214.10	214.10	55.02	43	545
N20G1P50U2-2	205.80	205.80	208.76	79	672
N20G1P50U2-3	210.22	210.22	44.02	34	543
N20G1P50U3-1	225.68	225.68	481.25	63	652
N20G1P50U3-2	208.86	208.86	17.89	13	293
N20G1P50U3-3	209.84	209.84	19.70	16	339
N20G1P75U1-1	214.62	214.62	32.40	43	451
N20G1P75U1-2	198.96	198.96	34.08	40	474
N20G1P75U1-3	212.02	212.02	7.60	22	423
N20G1P75U2-1	205.46	205.46	83.07	55	565
N20G1P75U2-2	225.30	225.30	314.99	71	714
N20G1P75U2-3	211.38	211.38	29.19	42	420
N20G1P75U3-1	209.20	209.20	26.05	32	399
N20G1P75U3-2	206.64	206.64	1.97	7	231
N20G1P75U3-3	209.60	209.60	1.57	5	222
N20G2P25U1-1	203.28	203.28	65.21	66	561
N20G2P25U1-2	215.58	215.58	77.60	57	672
N20G2P25U1-3	212.24	212.24	84.63	62	634
N20G2P25U2-1	217.68	217.68	55.30	43	532
N20G2P25U2-2	218.16	218.16	1624.73	116	989
N20G2P25U2-3	228.10	228.10	467.96	59	889
N20G2P25U3-1	221.50	221.50	152.75	41	624
N20G2P25U3-2	221.56	-	7200.00	182	1132
N20G2P25U3-3	220.64	220.64	907.96	67	803
N20G2P50U1-1	202.72	202.72	383.61	96	844
N20G2P50U1-2	198.30	198.30	445.96	91	823
N20G2P50U1-3	221.64	221.64	40.74	45	522
N20G2P50U2-1	230.54	230.54	7164.19	135	1113
N20G2P50U2-2	225.42	225.42	207.92	40	721
N20G2P50U2-3	219.46	219.46	211.38	63	678
N20G2P50U3-1	230.50	230.50	675.38	58	810
N20G2P50U3-2	239.40	283.92	7200.00	114	1115
N20G2P50U3-3	224.06	224.06	974.09	80	764
N20G2P75U1-1	202.58	202.58	1.61	6	209
N20G2P75U1-2	205.24	205.24	539.34	101	906
N20G2P75U1-3	205.44	205.44	559.73	119	857
N20G2P75U2-1	202.68	202.68	4.11	9	272
N20G2P75U2-2	224.58	224.58	5.41	15	349
N20G2P75U2-3	218.58	218.58	12.89	27	372
N20G2P75U3-1	220.34	220.38	7200.00	103	951
N20G2P75U3-2	220.46	220.48	7200.00	117	1017
N20G2P75U3-3	211.58	211.58	1.82	6	244
N20G3P25U1-1	221.18	221.18	102.53	58	663
N20G3P25U1-2	189.56	189.56	431.48	96	843
N20G3P25U1-3	189.26	189.26	785.55	78	884
N20G3P25U2-1	209.94	209.94	81.29	41	579
N20G3P25U2-2	216.78	226.26	7200.00	109	1289
N20G3P25U2-3	228.32	228.32	1721.03	70	1070
N20G3P25U3-1	213.94	213.94	6739.51	117	1125
N20G3P25U3-2	238.22	-	7200.00	101	1136
N20G3P25U3-3	233.82	270.04	7200.00	113	1095
N20G3P50U1-1	199.52	199.52	33.47	39	513
N20G3P50U1-2	193.92	193.92	821.79	126	848
N20G3P50U1-3	200.52	200.52	1868.88	169	1210
N20G3P50U2-1	198.72	198.72	414.70	58	713
N20G3P50U2-2	257.66	257.68	7200.00	115	1104
N20G3P50U2-3	210.14	210.14	1306.51	60	872
N20G3P50U3-1	235.62	235.64	7200.00	128	1100
N20G3P50U3-2	231.50	231.50	26.43	19	399
N20G3P50U3-3	248.74	248.74	23.44	19	435
N20G3P75U1-1	156.96	156.96	180.76	74	633
N20G3P75U1-2	208.54	208.54	451.58	97	844
N20G3P75U1-3	191.74	191.74	46.35	35	582
N20G3P75U2-1	226.54	226.54	379.16	57	647
N20G3P75U2-2	206.52	213.18	7200.00	75	942
N20G3P75U2-3	236.56	242.12	7200.00	133	1048
N20G3P75U3-1	214.70	215.32	7200.00	139	898
N20G3P75U3-2	234.04	234.04	3158.80	64	905
N20G3P75U3-3	234.84	234.84	529.05	53	671

Table C.3: Instances with 30 customers (N30).

Instance	Lower	Best	Runtime	Iterations	Points
N30G1P25U1-1	307.18	307.18	435.25	88	1165
N30G1P25U1-2	309.42	309.42	7077.08	231	2082
N30G1P25U1-3	310.52	310.52	2959.09	166	1767
N30G1P25U2-1	310.00	310.00	413.55	54	1093
N30G1P25U2-2	306.62	306.62	1418.79	96	1307
N30G1P25U2-3	321.04	321.04	1203.95	89	1306
N30G1P25U3-1	310.60	310.60	1040.41	96	995
N30G1P25U3-2	319.32	319.32	1956.07	68	1189
N30G1P25U3-3	304.54	313.70	7200.00	187	1529
N30G1P50U1-1	311.34	311.34	5647.27	219	1887
N30G1P50U1-2	301.48	301.48	789.43	120	1326
N30G1P50U1-3	310.54	310.54	4729.12	220	1613
N30G1P50U2-1	315.32	315.32	231.52	49	882
N30G1P50U2-2	320.22	320.22	552.07	60	1089
N30G1P50U2-3	312.18	312.18	300.87	49	1035
N30G1P50U3-1	313.48	313.48	83.43	23	638
N30G1P50U3-2	323.10	323.10	2084.87	90	1116
N30G1P50U3-3	309.22	309.22	29.26	11	461
N30G1P75U1-1	306.68	306.68	487.31	86	1088
N30G1P75U1-2	311.74	311.74	2261.30	163	1425
N30G1P75U1-3	296.98	296.98	422.31	84	1018
N30G1P75U2-1	312.36	312.36	19.98	17	507
N30G1P75U2-2	316.02	316.02	4233.20	187	1360
N30G1P75U2-3	316.24	316.24	3370.19	128	1207
N30G1P75U3-1	318.20	318.20	1166.61	78	891
N30G1P75U3-2	304.88	304.90	7200.00	149	1208
N30G1P75U3-3	308.38	308.38	10.12	10	389
N30G2P25U1-1	295.90	295.90	2335.74	126	1586
N30G2P25U1-2	319.74	319.74	733.21	86	1398
N30G2P25U1-3	314.22	314.22	579.88	90	1244
N30G2P25U2-1	309.66	309.66	1562.17	94	1388
N30G2P25U2-2	338.74	346.40	7200.00	113	1547
N30G2P25U2-3	309.40	309.40	1750.36	104	1346
N30G2P25U3-1	317.88	324.90	7200.00	104	1443
N30G2P25U3-2	340.06	-	7200.00	83	1435
N30G2P25U3-3	319.64	-	7200.00	76	1484
N30G2P50U1-1	286.36	286.36	730.75	95	1192
N30G2P50U1-2	330.86	330.86	106.11	38	915
N30G2P50U1-3	298.54	298.54	1463.70	111	1390
N30G2P50U2-1	322.46	322.46	2247.04	102	1182
N30G2P50U2-2	330.66	330.66	481.76	51	1008
N30G2P50U2-3	301.02	308.02	7200.00	114	1311
N30G2P50U3-1	331.76	331.76	1549.20	67	986
N30G2P50U3-2	328.10	335.40	7200.00	74	1370
N30G2P50U3-3	314.76	353.38	7200.00	88	1330
N30G2P75U1-1	304.86	304.86	2100.89	134	1386
N30G2P75U1-2	302.94	302.94	1044.95	117	1132
N30G2P75U1-3	304.52	304.52	151.35	47	877
N30G2P75U2-1	310.26	310.26	3083.55	92	1207
N30G2P75U2-2	312.82	312.82	148.60	41	843
N30G2P75U2-3	315.44	315.44	953.23	76	1005
N30G2P75U3-1	320.60	320.72	7200.00	125	1192
N30G2P75U3-2	318.20	318.20	72.45	19	607
N30G2P75U3-3	315.78	-	7200.00	105	1167
N30G3P25U1-1	298.04	298.04	1593.19	94	1496
N30G3P25U1-2	311.88	311.88	143.00	52	914
N30G3P25U1-3	302.08	302.08	3123.85	105	1569
N30G3P25U2-1	326.92	-	7200.00	86	1602
N30G3P25U2-2	330.38	330.38	6575.17	128	1543
N30G3P25U2-3	325.92	325.92	5162.23	113	1506
N30G3P25U3-1	331.44	-	7200.00	69	1452
N30G3P25U3-2	342.44	-	7200.00	82	1380
N30G3P25U3-3	346.22	-	7200.00	63	1304
N30G3P50U1-1	273.32	273.32	2940.29	113	1452
N30G3P50U1-2	302.80	302.80	768.84	61	1244
N30G3P50U1-3	327.88	327.88	291.60	41	913
N30G3P50U2-1	339.96	339.96	2091.10	82	1129
N30G3P50U2-2	348.34	348.34	300.00	45	941
N30G3P50U2-3	349.44	349.44	3362.65	48	1240
N30G3P50U3-1	347.56	369.54	7200.00	60	1197
N30G3P50U3-2	334.46	-	7200.00	52	1144
N30G3P50U3-3	339.98	348.08	7200.00	92	1269
N30G3P75U1-1	283.22	283.22	2568.24	82	1263
N30G3P75U1-2	318.98	318.98	266.97	51	1056
N30G3P75U1-3	272.84	-	7200.00	114	1596
N30G3P75U2-1	332.68	362.10	7200.00	78	1308
N30G3P75U2-2	342.06	342.06	6811.06	104	1408
N30G3P75U2-3	321.84	327.30	7200.00	101	1422
N30G3P75U3-1	330.64	380.78	7200.00	59	1233
N30G3P75U3-2	357.79	-	7200.00	109	1348
N30G3P75U3-3	330.62	375.94	7200.00	65	1371

Appendix 2 – Iterations and Status Tables

In Table C.4 the average number of iterations are computed for each instance type.

Table C.4: Average number of iterations.

		N10			N20			N30		
		U1	U2	U3	U1	U2	U3	U1	U2	U3
G1	P25	38	30	5	87	52	32	162	80	117
	P50	31	14	11	75	52	31	186	53	41
	P75	35	23	20	35	56	15	111	111	79
G2	P25	23	23	46	62	73	97	101	104	88
	P50	34	17	6	77	79	84	81	89	76
	P75	21	17	32	75	17	75	99	70	83
G3	P25	30	44	18	77	73	110	84	109	71
	P50	21	14	35	111	78	55	72	58	68
	P75	22	15	57	69	88	85	82	94	78

Tables C.5, C.6 and C.7 shows the number of instances with status Optimal, Feasible or No Solution after 2 hours for the number of customers and usage, geographical distribution and percentage.

Table C.5: Instance status for number of customers and usage rate

Customers	10			20			30		
Usage	U1	U2	U3	U1	U2	U3	U1	U2	U3
Optimal	27	27	26	27	23	19	26	22	9
Feasible	0	0	1	0	4	6	0	4	10
No Solution	0	0	0	0	0	2	1	1	8

Table C.6: Instance status for number of customers and geographical distribution

Customers	10			20			30		
Distribution	G1	G2	G3	G1	G2	G3	G1	G2	G3
Optimal	27	27	26	27	23	19	25	18	14
Feasible	0	0	1	0	3	7	2	6	6
No Solution	0	0	0	0	1	1	0	3	7

Table C.7: Instance status for number of customers and percentage N_ℓ customers

Customers	10			20			30		
Percentage	P25	P50	P75	P25	P50	P75	P25	P50	P75
Optimal	27	27	26	23	24	22	18	21	18
Feasible	0	0	1	2	3	5	3	5	6
No Solution	0	0	0	2	0	0	6	1	3

Appendix 3 – CIRP-OB Instances and Results for [72]

In [72] randomly generated instances are solved using lower and upper bound models. We adapt those instances to the CIRP-OB and find new values for the number of vehicles, using the same methodology is used for finding a number of vehicles for CIRP instances. We run the lower bound model and the upper bound model with a homogeneous discretization $\Delta = 3$ and $\Delta = 1$, respectively, for 2 hours. The optimality gap is reported in Tables C.8 and C.9. Several instances do not have a provable optimal solution, specially random instances with more than 10 customers.

Table C.8: Optimality gap (%) for clustered instances using LOWER(3) and UPPER(1) (2 hrs).

		Usage Level								
		U1			U2			U3		
		Capacity			**Capacity**			**Capacity**		
		Q1	Q2	Q3	Q1	Q2	Q3	Q1	Q2	Q3
Customers	5	0.00	0.00	18.85	0.00	0.00	0.00	0.00	0.00	0.00
	7	0.00	0.00	0.00	0.00	0.00	0.00	0.00	8.67	10.66
	10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	13.45	15.33
	12	0.00	6.09	14.61	0.00	5.38	6.09	3.90	0.00	0.00
	15	0.00	10.28	17.67	0.00	4.54	4.97	3.28	0.00	0.00

Table C.9: Optimality gap (%) for random instances using LOWER(3) and UPPER(1) (2 hrs).

		Usage Level								
		U1			U2			U3		
		Capacity			**Capacity**			**Capacity**		
		Q1	Q2	Q3	Q1	Q2	Q3	Q1	Q2	Q3
Customers	5	0.00	15.80	17.50	0.00	0.00	0.00	0.00	0.00	0.00
	7	0.00	0.00	6.28	0.00	0.00	0.00	0.00	0.00	0.00
	10	12.79	3.03	8.27	2.26	12.79	21.89	1.86	2.37	8.14
	12	6.25	7.80	15.27	1.86	10.38	7.80	1.58	5.89	6.51
	15	8.97	6.54	10.45	4.68	17.51	15.28	4.08	8.17	13.14

The DDD algorithm runs for the same instances for up to 2 hours. In this case, all the instances have an optimal solution. In Tables C.10 and C.11 the running time

is shown for each instance.

Table C.10: Running time (seconds) for clustered instances.

		Usage Level								
		U1			U2			U3		
		Capacity			**Capacity**			**Capacity**		
		Q1	Q2	Q3	Q1	Q2	Q3	Q1	Q2	Q3
Customers	5	0.19	0.19	0.73	0.20	0.19	0.19	0.22	0.20	0.18
	7	0.21	0.20	0.20	0.22	0.21	0.20	0.24	0.21	0.21
	10	1.14	0.99	1.11	1.14	1.08	1.07	1.33	1.09	1.05
	12	1.20	1.04	2.94	1.28	1.31	1.05	1.28	1.22	1.12
	15	1.29	1.24	4.36	0.33	0.32	0.24	0.37	0.29	0.25

Table C.11: Running time (seconds) for random instances.

		Usage Level								
		U1			U2			U3		
		Capacity			**Capacity**			**Capacity**		
		Q1	Q2	Q3	Q1	Q2	Q3	Q1	Q2	Q3
Customers	5	0.24	49.42	44.79	0.25	0.21	0.23	0.23	0.23	0.21
	7	0.26	0.21	5.34	0.25	0.23	0.22	0.26	0.23	0.21
	10	141.04	33.81	102.25	0.26	4.84	262.97	0.34	0.25	6.72
	12	0.28	101.49	115.75	0.32	71.58	1.98	0.34	15.02	88.13
	15	0.31	596.33	645.91	0.34	434.38	1919.30	0.58	39.00	1159.75

REFERENCES

[1] G. B. Dantzig and J. H. Ramser, "The truck dispatching problem," *Management science*, vol. 6, no. 1, pp. 80–91, 1959.

[2] W. Bell, L. Dalberto, M. Fisher, A. Greenfield, R. Jaikumar, P. Kedia, R. Mack, and P. Prutzman, "Improving the distribution of industrial gases with an online computerized routing and scheduling optimizer.," *Interfaces*, vol. 13, no. 6, pp. 4–23, 1983.

[3] G. Laporte, "Fifty years of vehicle routing," *Transportation science*, vol. 43, no. 4, pp. 408–416, 2009.

[4] A. Campbell, L. Clarke, A. Kleywegt, and M. Savelsbergh, "The inventory routing problem," in *Fleet management and logistics*, T. Crainic and G. Laport, Eds., Norwell, MA: Kluwer Academic Publishers, 1998, pp. 95–113.

[5] L. C. Coelho, J. F. Cordeau, and G. Laporte, "Thirty years of inventory routing," *Transportation science*, vol. 48, no. 1, pp. 1–19, 2014.

[6] V. Pillac, M. Gendreau, C. Guéret, and A. L. Medaglia, "A review of dynamic vehicle routing problems," *European journal of operational research*, vol. 225, no. 1, pp. 1–11, 2013.

[7] A. M. Campbell and B. W. Thomas, "Challenges and advances in a priori routing," in *The Vehicle Routing Problem: Latest Advances and New Challenges*, Springer, 2008, pp. 123–142.

[8] N. Boland, M. Hewitt, L. Marshall, and M. Savelsbergh, "The continuous-time service network design problem," *Operations research*, vol. 65, no. 5, pp. 1303–1321, 2017.

[9] E. He, N. Boland, G. Nemhauser, and M. Savelsbergh, "Dynamic discretization discovery algorithms for time-dependent shortest path problems," *Optimization online*, vol. 7082, 2019.

[10] D. M. Vu, N Boland, M Hewitt, and M. Savelsbergh, "Solving time dependent traveling salesman problems with time windows," *Optimization online*, vol. 6640, 2018.

[11] J. F. Cordeau, G. Laporte, M. W. P. Savelsbergh, and D. Vigo, "Vehicle routing," *Transportation, handbooks in operations research and management science*, vol. 14, pp. 367–428, 2006.

[12] B. L. Golden, S. Raghavan, and E. A. Wasil, Eds., *The Vehicle Routing Problem: Latest Advances and New Challenges*. Springer, 2008.

[13] P. Toth and D. Vigo, *Vehicle routing: Problems, methods, and applications*. SIAM, 2014, vol. 18.

[14] P. Jaillet, "Probabilistic traveling salesman problems," PhD thesis, Massachusetts Institute of Technology, 1985.

[15] P. Jaillet, "A priori solution of a traveling salesman problem in which a random subset of the customers are visited," *Operations research*, vol. 36, no. 6, pp. 929–936, 1988.

[16] M. Gendreau, G. Laporte, and R. Séguin, "An exact algorithm for the vehicle routing problem with stochastic demands and customers," *Transportation science*, vol. 29, no. 2, pp. 143–155, 1995.

[17] D. J Bertsimas, "A vehicle routing problem with stochastic demand," *Operations research*, vol. 40, no. 3, pp. 574–585, 1992.

[18] J. J. Bartholdi III, L. K. Platzman, R. L. Collins, and W. H. Warden III, "A minimal technology routing system for meals on wheels," *Interfaces*, vol. 13, no. 3, pp. 1–8, 1983.

[19] J. Cordeau, G. Laporte, F. Pasin, and S. Ropke, "Scheduling technicians and tasks in a telecommunications company," *Journal of scheduling*, vol. 13, no. 4, pp. 393–409, 2010.

[20] H. Hashimoto, S. Boussier, M. Vasquez, and C. Wilbaut, "A grasp-based approach for technicians and interventions scheduling for telecommunications," *Annals of operations research*, vol. 183, no. 1, pp. 143–161, 2011.

[21] V. Pillac, C. Gueret, and A. Medaglia, "A parallel matheuristic for the technician routing and scheduling problem," *Optimization letters*, vol. 7, no. 7, pp. 1525–1535, 2013.

[22] L. Talarico, K. Sörensen, and J. Springael, "Metaheuristics for the risk-constrained cash-in-transit vehicle routing problem," *European journal of operational research*, vol. 244, no. 2, pp. 457–470, 2015.

[23] R. Bent and P. van Hentenryck, "Scenario-based planning for partially dynamic vehicle routing with stochastic customers," *Operations research*, vol. 52, no. 6, pp. 977–987, 2004.

[24] A. Larsen, O. B. G. Madsen, and M. M. Solomon, "The a priori dynamic traveling salesman problem with time windows," *Transportation science*, vol. 38, no. 4, pp. 459–472, 2004.

[25] S. Voccia, A. Campbell, and B. Thomas, "The same-day delivery problem for online purchases," *Published online on transportation science*, pp. 1–18, 2015.

[26] D. Bertsekas and D. Castanon, "Rollout algorithms for stochastic scheduling problems," *Journal of heuristics*, vol. 5, no. 1, pp. 89–108, 1999.

[27] D. Bertsekas, J. N Tsitsiklis, and C. Wu, "Rollout algorithms for combinatorial optimization," *Journal of heuristics*, vol. 3, no. 3, pp. 245–262, 1997.

[28] J. C Goodson, B. W Thomas, and J. W Ohlmann, "A rollout algorithm framework for heuristic solutions to finite-horizon stochastic dynamic programs," *European journal of operational research*, vol. 258, no. 1, pp. 216–229, 2017.

[29] M. W. Ulmer, D. C. Mattfeld, M. Hennig, and J. C. Goodson, "A rollout algorithm for vehicle routing with stochastic customer requests," in *Logistics management*, Springer, 2016, pp. 217–227.

[30] M. Klapp, A. Erera, and A. Toriello, "The dynamic dispatch waves problem for same-day delivery," *Under review*, pp. 1–26, 2017.

[31] M. "Klapp, A. Erera, and A. Toriello, "The one-dimensional dynamic dispatch waves problem," *Transportation science*, vol. 52, no. 2, pp. 402–415, 2016.

[32] M. Klapp, "Dynamic Optimization for Same-Day Delivery Operations," PhD thesis, Georgia Institute of Technology, 2016.

[33] G. Laporte, F. V. Louveaux, and H. Mercure, "A priori optimization of the probabilistic traveling salesman problem," *Operations research*, vol. 42, no. 3, pp. 543–549, 1994.

[34] G. Laporte and F. Louveaux, "The integer l-shaped method for stochastic integer programs with complete recourse," *Operations research letters*, vol. 13, no. 3, pp. 133–142, 1993.

[35] A. M. Campbell and B. W. Thomas, "Probabilistic traveling salesman problem with deadlines," *Transportation science*, vol. 42, no. 1, pp. 1–21, 2008.

[36] S. Voccia, A. M. Campbell, and B. W. Thomas, "The probabilistic traveling salesman problem with time windows," *Euro journal on transportation and logistics*, pp. 1–19, 2012.

[37] G. Desaulniers, J. Desrosiers, and M. M. Solomon, *Column generation*. Springer Science & Business Media, 2006, vol. 5.

[38] M. M. Solomon, "Algorithms for the vehicle routing and scheduling problems with time window constraints," *Operations research*, vol. 35, no. 2, pp. 254–265, 1987.

[39] D. J. Bertsimas, P. Jaillet, and A. R. Odoni, "A priori optimization," *Operations research*, vol. 38, no. 6, pp. 1019–1033, 1990.

[40] P. Jaillet, "Analysis of probabilistic combinatorial optimization problems in euclidean spaces," *Mathematics of operations research*, vol. 18, no. 1, pp. 51–70, 1993.

[41] F. A. Tillman, "The multiple terminal delivery problem with probabilistic demands," *Transportation science*, vol. 3, no. 3, pp. 192–204, 1969.

[42] G. Laporte, F. Louveaux, and H. Mercure, "The vehicle routing problem with stochastic travel times," *Transportation science*, vol. 26, no. 3, pp. 161–170, 1992.

[43] T. Leipälä, "On the solutions of stochastic traveling salesman problems," *European journal of operational research*, vol. 2, no. 4, pp. 291–297, 1978.

[44] D. J. Bertsimas, "Probabilistic combinatorial optimization problems," PhD thesis, Massachusetts Institute of Technology, 1988.

[45] O. Berman and D. Simchi-Levi, "Finding the optimal a priori tour and location of a traveling salesman with nonhomogeneous customers," *Transportation science*, vol. 22, no. 2, pp. 148–154, 1988.

[46] D. J. Bertsimas and L. H. Howell, "Further results on the probabilistic traveling salesman problem," *European journal of operational research*, vol. 65, no. 1, pp. 68–95, 1993.

[47] M. Gendreau, G. Laporte, and R. Seguin, "Stochastic vehicle routing," *European journal of operational research*, vol. 88, no. 1, pp. 3–12, 1996.

[48] H. Tang and E. Miller-Hooks, "Approximate procedures for probabilistic traveling salesperson problem," *Transportation research record: Journal of the transportation research board*, no. 1882, pp. 27–36, 2004.

[49] J. Branke and M. Guntsch, "Solving the probabilistic tsp with ant colony optimization," *Journal of mathematical modelling and algorithms*, vol. 3, no. 4, pp. 403–425, 2005.

[50] C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, and P. H. Vance, "Branch-and-price: Column generation for solving huge integer programs," *Operations research*, vol. 46, pp. 316–329, 1998.

[51] M. Desrochers, J. Desrosiers, and M. M. Solomon, "A new optimization algorithm for the vehicle routing problem with time windows," *Operations research*, vol. 40, no. 2, pp. 342–354, 1992.

[52] J. Desrosiers and M. E. Lübbecke, "A primer in column generation," in *Column generation*, Springer US, 2005, pp. 1–32.

[53] C. H. Christiansen and J. Lysgaard, "A branch-and-price algorithm for the capacitated vehicle routing problem with stochastic demands," *Operations research letters*, vol. 35, no. 6, pp. 773–781, 2007.

[54] T. Dinh, R. Fukasawa, and J. Luedtke, "Exact algorithms for the chance-constrained vehicle routing problem," in *International conference on integer programming and combinatorial optimization*, Springer, 2016, pp. 89–101.

[55] C. Gauvin, G. Desaulniers, and M. Gendreau, "A branch-cut-and-price algorithm for the vehicle routing problem with stochastic demands," *Computers & operations research*, vol. 50, pp. 141–153, 2014.

[56] F. Errico, G. Desaulniers, M. Gendreau, and L.-M. Rousseau, "A priori optimization with recourse for the vehicle routing problem with hard time windows and stochastic service times," *European journal of operational research*, vol. 249, pp. 55–66, 2016.

[57] D. Taş, M. Gendreau, N. Dellaert, T. Van Woensel, and A. De Kok, "Vehicle routing with soft time windows and stochastic travel times: A column generation and branch-and-price solution approach," *European journal of operational research*, vol. 236, no. 3, pp. 789–799, 2014.

[58] S. Irnich and D. Villeneuve, "The shortest-path problem with resource constraints and k-cycle elimination for $k \geq 3$," *Informs journal on computing*, vol. 18, pp. 391–406, 2006.

[59] M. Dror, "Note on the complexity of the shortest path models for column generation in vrptw," *Operations research*, vol. 42, no. 5, pp. 977–978, 1994.

[60] R. Baldacci, A. Mingozzi, and R. Roberti, "New route relaxation and pricing strategies for the vehicle routing problem," *Operations research*, vol. 59, no. 5, pp. 1269–1283, 2011.

[61] M. E Lubbecke and J. Desrosiers, "Selected topics in column generation," *Operations research*, vol. 53, no. 6, pp. 1007–1023, 2005.

[62] L. Bertazzi, M. Savelsbergh, and M. G. Speranza, "Inventory routing," in *The vehicle routing problem: Latest advances and new challenges*, B. Golden, S. Raghavan, and E. Wasil, Eds., Boston, MA: Springer US, 2008, pp. 49–72.

[63] A. Campbell, L. Clarke, and M. Savelsbergh, "Inventory routing in practice," in *The vehicle routing problem*, P. Toth and D. Vigo, Eds., Philadelphia, PA, USA: Society for Industrial, Applied Mathematics, monographs on discrete mathematics, and applications, 2002, pp. 309–330.

[64] A. Campbell and M. Savelsbergh, "A decomposition approach for the inventory routing problem," *Transportation science*, vol. 38, pp. 488–502, 2004.

[65] A. Campbell and M. Savelsbergh", "Delivery volume optimization," *Transportation science*, vol. 38, pp. 210–223, 2004.

[66] L. C. Coelho and G. Laporte, "The exact solution of several classes of inventory-routing problems," *Computers & operations research*, vol. 40, no. 2, pp. 558 – 565, 2013.

[67] L. C. Coelho and G. Laporte", "Improved solutions for inventory-routing problems through valid inequalities and input ordering," *International journal of production economics*, vol. 155, no. C, pp. 391 –397, 2014, Celebrating a century of the economic order quantity model.

[68] P. Avella, M. Boccia, and L. Wolsey, "Single-period cutting planes for inventory routing problems," *Transportation science*, vol. 52, no. 3, pp. 497–508, 2017.

[69] G. Desaulniers, J. Rakke, and L. C. Coelho, "A branch-price-and-cut algorithm for the inventory-routing problem," *Transportation science*, vol. 50, pp. 1060–1076, 2015.

[70] J.-H. Song and M. Savelsbergh, "Performance measurement for inventory routing," *Transportation science*, vol. 41, no. 1, pp. 44–54, 2007.

[71] C. Archetti, L. Bertazzi, G. Laporte, and M. G. Speranza, "A branch-and-cut algorithm for a vendor-managed inventory-routing problem," *Transportation science*, vol. 41, no. 3, pp. 382–391, 2007.

[72] F. Lagos, N. Boland, and M. Savelsbergh, "The continuous time inventory routing problem," *Optimization-online*, 2018.

[73] A. J. Kleywegt, V. S. Nori, and M. W. Savelsbergh, "The stochastic inventory routing problem with direct deliveries," *Transportation science*, vol. 36, no. 1, pp. 94–118, 2002.

[74] L. Bertazzi, "Analysis of direct shipping policies in an inventory-routing problem with discrete shipping times," *Management science*, vol. 54, no. 4, pp. 748–762, 2008.

[75] G. Gallego and D. Simchi-Levi, "Rejoinder to âăija note on bounds for direct shipping costsâăi," *Management science*, vol. 40, no. 10, pp. 1393–1393, 1994.

[76] J. Li, H. Chen, and F. Chu, "Performance evaluation of distribution strategies for the inventory routing problem," *European journal of operational research*, vol. 202, no. 2, pp. 412–419, 2010.

[77] G. Gallego and D. Simchi-Levi, "On the effectiveness of direct shipping strategy for the one-warehouse multi-retailer r-systems," *Management science*, vol. 36, no. 2, pp. 240–243, 1990.